[1]

Method for determining the position of a processor unit from an adjacent processor unit in a processor array, the method relates particularly to TFT type displays and a method for detection of faulty pixels

Patent number: DE10158784
Publication date: 2003-08-07
Inventor: JUNG STEFAN (DE); BUCHMEIER ANTON (DE); STOEHR ANNELIE (DE); STURM THOMAS (DE)
Applicant: INFINEON TECHNOLOGIES AG (DE)
Classification:
- international: ***G06F15/80; G09G3/20;*** **G06F15/76; G09G3/20;** (IPC1-7): G06F15/173; G06F15/80; G09G3/36
- european: G06F15/80; G09G3/20
Application number: DE20011058784 20011130
Priority number(s): DE20011058784 20011130

Abstract of **DE10158784**
Within an array each processor is connected to adjacent processors by a bi-directional communication interface. Messages are exchanged between the units to determine the separation of a unit from a reference point. Each message contains separation information, i.e. the separation of the unit receiving the message from the reference position or the separation of the unit sending the message from the reference position. Each unit is able to determine or store its separation from the reference position.

(19) BUNDESREPUBLIK DEUTSCHLAND

DEUTSCHES PATENT- UND MARKENAMT

(12) **Offenlegungsschrift**

(10) **DE 101 58 784 A 1**

(51) Int. Cl.[7]:
G 06 F 15/173
G 06 F 15/80
G 09 G 3/36

(21) Aktenzeichen: 101 58 784.8
(22) Anmeldetag: 30. 11. 2001
(43) Offenlegungstag: 7. 8. 2003

(71) Anmelder:
Infineon Technologies AG, 81669 München, DE

(74) Vertreter:
Viering, Jentschura & Partner, 80538 München

(72) Erfinder:
Jung, Stefan, Dr., 80469 München, DE; Buchmeier, Anton, Dr., 85521 Ottobrunn, DE; Stöhr, Annelie, Dr., 80804 München, DE; Sturm, Thomas, Dr., 81929 München, DE

(56) Entgegenhaltungen:

DE	199 50 839 A1
DE	197 10 855 A1
DE	196 54 440 A1
DE	196 53 288 A1
DE	42 03 276 A1
DE	38 37 313 A1
DE	36 33 708 A1
DE	201 08 797 U1
US	58 80 705 A
US	56 44 327 A

Die folgenden Angaben sind den vom Anmelder eingereichten Unterlagen entnommen

Prüfungsantrag gem. § 44 PatG ist gestellt

(54) Verfahren zum Bestimmen eines Abstands von Prozessoreinheiten zu mindestens einer Referenzposition in einer Prozessor-Anordnung und Prozessor-Anordnung

(57) Eine Prozessor-Anordnung weist eine Vielzahl von Prozessoreinheiten auf,

- wobei jede Prozessoreinheit über eine bidirektionale Kommunikationsschnittstelle mit mindestens einer benachbarten Prozessoreinheit gekoppelt ist und wobei zum Ermitteln des jeweiligen Abstands einer Prozessoreinheit der Prozessor-Anordnung von einer Referenzposition Nachrichten ausgetauscht werden zwischen einander benachbarten Prozessoreinheiten,
- wobei jede Nachricht eine Abstandsinformation enthält, welche den Abstand einer die Nachricht sendenden Prozessoreinheit oder den Abstand einer die Nachricht empfangenden Prozessoreinheit von der Referenzposition angibt, und
- wobei jede Prozessoreinheit derart eingerichtet ist, dass aus der Abstandsinformation einer empfangenen Nachricht der eigene Abstand zu der Referenzposition ermittelbar ist oder speicherbar ist.

DE 101 58 784 A1

DE 101 58 784 A1

Beschreibung

[0001] Die Erfindung betrifft ein Verfahren zum Bestimmen eines Abstands von Prozessoreinheiten zu mindestens einer Referenzposition in einer Prozessor-Anordnung sowie eine Prozessor-Anordnung.

[0002] Wenn eine Prozessor-Anordnung mit einer Vielzahl von Prozessoren derart hergestellt worden ist, dass die örtliche Position der einzelnen Prozessoren innerhalb der Prozessor-Anordnung nicht bekannt ist, können die einzelnen Prozessoren nicht individuell adressiert werden.

[0003] Vor dem Einsatz der Prozessor-Anordnung müssen die Prozessoren jedoch organisiert werden, damit die örtliche Position der einzelnen Prozessoren innerhalb der Prozessor-Anordnung ermittelt werden kann und damit die Prozessoren innerhalb der Prozessor-Anordnung individuell adressieren zu können.

[0004] Die Organisation sollte selbst bei auftretenden Fehlern im Rahmen der Herstellung der Prozessor-Anordnung oder bei späteren Ausfällen eines oder mehrerer Prozessoren oder einer oder mehrerer Verbindungen zwischen den Prozessoren durchführbar sein.

[0005] Ein Beispiel einer solchen Prozessor-Anordnung ist in einer Pixel-Anordnung zu sehen, wobei jedem Pixel der Pixel-Anordnung ein Prozessor zugeordnet ist, der das Pixel ansteuern kann. Das Pixel kann als ein bildgebendes Element ausgestaltet sein oder auch als ein Sensorelement, so dass die Pixel-Anordnung als ein Anzeigeeinheit oder als ein Sensorfeld ausgestaltet sein kann.

[0006] In einer Pixel-Anordnung mit einer Vielzahl von Pixeln, insbesondere bei einer großflächigen Punkt-Matrix-Anzeigeeinheit oder einem großflächigen Sensorfeld treten oftmals erhebliche Probleme auf.

[0007] Eine solche Pixel-Anordnung weist beispielsweise für den Fall, dass sie als sogenannte "elektronische Zeitung" eingerichtet ist, pro Seite mehrere Millionen Bildpunkte, das heißt Pixel, auf.

[0008] Im Folgenden wird unter einer Pixel-Anordnung eine homogene Pixelmatrix verstanden, welche von deren Rändern aus angesteuert bzw. adressiert wird. Eine solche Anzeigematrix oder eine Sensormatrix, das heißt allgemein eine solche Pixel-Anordnung, weist beispielsweise aktive nicht-lineare Auswahleinrichtungen auf wie beispielsweise Dünnfilmtransistoren (Thin Film Transistors, TFTs) in Flüssigkristall-Anzeigeeinheiten (Liquid Crystal Displays, LCDs), welche in ihrer Größe bzw. Dimensionalität begrenzt sind durch die Eigenschaften der Dünnfilmtransistoren sowie durch die parasitären Widerstände der Datenleitungen, welche zur Übertragung der Signale von und zu den einzelnen Pixeln verwendet werden.

[0009] Die Dünnfilmtransistoren müssen in der Praxis ein Stromverhältnis zwischen dem Strom, der bei einem Einschalt-Zustand fließt (I_{on}) zu dem Strom, der bei einem Ausschalt-Zustand fließt (I_{off}) von 10^5 bis 10^6 aufweisen. Hohe elektrische Widerstände der sehr dünnen und sehr langen Datenleitungen in einer solchen Pixel-Anordnung sowie ein niedriger Strom bei dem Einschalt-Zustand begrenzen die Zugriffszeit auf einzelne Zeilen bzw. Spalten der üblicherweise in einer Matrix angeordneten Pixel in der Pixel-Anordnung.

[0010] Dies führt zu einer nur sehr langsamen elektrischen Aufladung der einzelnen Pixelelemente. Ferner führt ein zu hoher Strom bei einem Ausschalt-Zustand der Pixel-Anordnung bei einem zeilenweisen oder spaltenweisen Abrastern der matrixförmigen Pixel-Anordnung zu einem Ladungsverlust elektrischer Ladungen in den Pixelkondensatoren, wenn diese gerade nicht ausgewählt sind. Aus diesem Grund darf ein Ansteuerungszyklus, das heißt ein Zeitintervall, zwischen denen jeweils eine Ansteuerung der einzelnen Pixelelemente in der Pixel-Anordnung erfolgt, nicht zu lang werden.

[0011] Die oben genannten Probleme sind beispielsweise in [1] und in [2] im Zusammenhang mit dem Entwurf von Flüssigkristall-Anzeigeeinheiten bzw. elektrophoretischen Anzeigeeinheiten mit Dünnfilmtransistoren als Ansteuertransistoren erläutert.

[0012] Ferner ist der Herstellungsprozess von Flüssigkristall-Flachbildschirmen extrem aufwändig und störanfällig.

[0013] Die oben beschriebenen Probleme verschärfen sich noch bei großflächigen und flexiblen Displays, das heißt bei großflächigen und biegsamen Anzeigeeinheiten wie beispielsweise bei einer elektronischen Zeitung, insbesondere für den Fall, dass für die Schaltelemente kostengünstige Herstellungsverfahren wie beispielsweise Druckverfahren zum Herstellen von druckbaren Transistoren eingesetzt werden, beispielsweise also im Bereich der Polymerelektronik. Solche Transistoren besitzen typischerweise ein Verhältnis von I_{on}/I_{off} von 10^4 bis 10^5. Jedenfalls sind die Transistoren, welche in Polymerelektronik realisiert sind, in ihren Eigenschaften noch schlechter als konventionelle Dünnfilmtransistoren auf Siliziumbasis. Solche Polymerelektronik-Transistoren eignen sich jedoch nur für Anzeigeeinheiten mit einigen Hundert Bildzeilen bzw. Bildspalten.

[0014] Ein besonderes Problem ist noch darin zu sehen, dass die Ausbeute bei der Herstellung einer Pixel-Anordnung aufgrund der sehr großen Fläche geringer wird. Dies ist auf die höhere Fehlerwahrscheinlichkeit bei einem solchen Herstellungsprozess zurück zu führen. Anders ausgedrückt bedeutet dies, dass die Anzahl fehlerhafter Bildpunkte bzw. Bildbereiche oder fehlerhafter Sensorpunkte, das heißt Sensorelemente größer wird mit größer werdender Fläche der Pixel-Anordnung.

[0015] In der Handhabung einer sehr dünnen, flexiblen und eventuell großflächigen Anzeigeeinheit ist auch die Wahrscheinlichkeit des Auftretens von neuen Defekten bzw. Schäden während der Handhabung hoch. Bei einer konventionellen Adressierung einer matrixförmigen Pixel-Anordnung würde ein Defekt in der Matrix sofort zu einem Zeilenfehler bzw. Spaltenfehler führen oder es würde sogar ein ganzer Bereich der Pixel-Anordnung ausfallen.

[0016] Die oben beschriebenen Probleme wurden versucht gemäß dem Stand der Technik dadurch zu lösen, dass die Eigenschaften der Auswahltransistoren, das heißt insbesondere das Verhältnis von I_{on} zu I_{off} verbessert wurden sowie durch verbesserte Herstellungsverfahren. Ein Schutz gegen mechanische Belastung eines Flachbildschirms wurde durch ein entsprechendes Gehäuse oder aber durch ein spezielles Packaging versucht zu erreichen.

[0017] Der Erfindung liegt das Problem zugrunde, ein Verfahren zum Bestimmen eines Abstandes von Prozessoren zu mindestens einer Referenzposition in einer Prozessor-Anordnung sowie eine Prozessor-Anordnung anzugeben, bei dem bzw. bei der zumindest ein Teil der oben genannten Probleme des Standes der Technik reduziert werden.

[0018] Das Problem wird durch das Verfahren sowie durch die Pixel-Anordnung mit den Merkmalen gemäß den unabhängigen Patentansprüchen gelöst.

[0019] Bei einem Verfahren zum Bestimmen eines Abstandes von Prozessoreinheiten zu mindestens einer Referenzposition in einer Prozessor-Anordnung mit einer Vielzahl von Prozessoreinheiten ist jede Prozessoreinheit über eine bidirektionale Kommunikationsschnittstelle mit mindestens einer ihr benachbarten Prozessoreinheit gekoppelt. Die Prozessoreinheiten tauschen untereinander, insbesondere zwischen einander unmittelbar benachbarten Prozessoreinheiten elektronische Nachrichten aus. Gemäß dem Verfahren wird eine erste Nachricht von einer ersten Prozessoreinheit, welche sich an der mindestens einen Referenzposition befindet, erzeugt. Die erste Nachricht enthält eine erste Abstandsinformation, welche den Abstand der ersten Prozessoreinheit oder den Abstand einer die erste Nachricht empfangenden zweiten Prozessoreinheit von der Referenzposition enthält. Die erste Nachricht wird von der ersten Prozessoreinheit zu der zweiten Prozessoreinheit übertragen. Abhängig von der ersten Abstandsinformation wird der Abstand der zweiten Prozessoreinheit von der Referenzposition ermittelt oder gespeichert. Von der zweiten Prozessoreinheit wird eine zweite Nachricht erzeugt, welche eine zweite Abstandsinformation enthält, welche den Abstand der zweiten Prozessoreinheit oder den Abstand einer die zweite Nachricht empfangenden dritten Prozessoreinheit von der Referenzposition enthält. Die zweite Nachricht wird von der zweiten Prozessoreinheit zu der dritten Prozessoreinheit übertragen. Vorzugsweise erfolgt die Übermittlung der jeweiligen Nachrichten immer über die bidirektionale Kommunikationsschnittstelle der jeweiligen einander unmittelbar benachbarten Prozessoreinheiten. Abhängig von der zweiten Abstandsinformation wird der Abstand der dritten Prozessoreinheit von der Referenzposition ermittelt oder gespeichert. Die Speicherung erfolgt vorzugsweise jeweils lokal in einem lokalen Speicher, der einer jeweiligen Prozessoreinheit zugeordnet ist. Die oben beschriebenen Verfahrensschritte werden für alle Prozessoreinheiten in der Pixel-Anordnung entsprechend in iterativer Weise durchgeführt.

[0020] Die Referenzposition kann grundsätzlich beliebig sein, vorzugsweise ist die Referenzposition eine Position, an der sich ein im Weiteren beschriebener Portalprozessor befindet, welcher die Prozessoreinheiten in der Prozessor-Anordnung ansteuert und die Kommunikation von außerhalb der Prozessor-Anordnung anstößt. Die Referenzposition kann ferner eine Position innerhalb der Prozessor-Anordnung sein, wobei in diesem Fall vorzugsweise eine Prozessoreinheit an der Referenzposition angeordnet und dieser zugeordnet ist. Vorzugsweise befindet sich in diesem Fall die Referenzposition am Rand, d. h. an der obersten oder untersten Zeile oder der linken oder rechten Spalte für den Fall, dass die Prozessoreinheiten in der Prozessor-Anordnung matrixförmig in Zeilen und Spalten angeordnet sind. Die Übertragung von Information in oder aus der Prozessor-Anordnung erfolgt vorzugsweise mittels des Portalprozessors ausschließlich über zumindest einen Teil der sich am Rand der Prozessor-Anordnung befindenden Prozessoreinheiten.

[0021] Anschaulich bedeutet diese Vorgehensweise, dass ausgehend von einer "Einleit-Prozessoreinheit" an der Referenzposition üblicherweise am Rand der Prozessor-Anordnung, das heißt an einem bezüglich der Prozessor-Anordnung äußeren Pixel, ein erster Abstand zugeordnet wird, beispielsweise der Abstandswert "1", womit angegeben wird, dass die Einleit-Prozessoreinheit einen Abstand "1" von dem Portalprozessor aufweist. Für den Fall, dass jeweils in der jeweiligen Nachricht der Abstand der die Nachricht sendenden Prozessoreinheit von der Referenzposition in die Nachricht eingefügt wird und an die die Nachricht zu empfangende Prozessoreinheit übertragen wird, wird von der ersten Prozessoreinheit der Abstandswert "1" zu der zweiten Prozessoreinheit in der ersten Nachricht übermittelt und von der zweiten Prozessoreinheit wird der empfangene Abstandswert um einen Wert "1" inkrementiert. Der inkrementierte Wert "2" wird nunmehr als aktualisierter zweiter Abstandswert der zweiten Prozessoreinheit gespeichert. Der zweite Abstandswert wird um einen Wert "1" inkrementiert und ein dritter Abstandswert wird erzeugt und an die dritte Prozessoreinheit übertragen und dort gespeichert. Die entsprechende Vorgehensweise wird für alle Prozessoreinheiten in entsprechender Weise durchgeführt und der einem Prozessor jeweils zugeordnete Abstandswert wird nach Empfang einer Nachricht mit einer Abstandsinformation immer dann aktualisiert, wenn der empfangene Abstandswert kleiner ist als der gespeicherte Abstandswert.

[0022] Eine Prozessor-Anordnung weist eine Vielzahl von Prozessoreinheiten auf. Jede Prozessoreinheit ist über eine bidirektionale Kommunikationsschnittstelle mit mindestens einer ihr benachbarten Prozessoreinheit gekoppelt. Zum Ermitteln des jeweiligen Abstands einer Prozessoreinheit der Prozessor-Anordnung von einer Referenzposition werden Nachrichten zwischen den jeweiligen Prozessoreinheiten ausgetauscht, vorzugsweise zwischen einander benachbarten Prozessoreinheiten, wobei jede Nachricht eine Abstandsinformation enthält, welche den Abstand einer die Nachricht sendenden Prozessoreinheit oder einer die Nachricht empfangenden Prozessoreinheit von der Referenzposition angibt (auch als Abstandswert bezeichnet) und wobei jede Prozessoreinheit derart eingerichtet ist, dass aus der Abstandsinformation einer empfangenen Nachricht der eigene Abstand zu der Referenzposition ermittelbar oder speicherbar ist.

[0023] Die Erfindung kann anschaulich darin gesehen werden, dass die gemäß dem Stand der Technik vorgesehene globale und direkte Prozessoransteuerung über Spalten- und Zeilenleitungen aufgegeben wird.

[0024] Aufgrund des Einsatzes lediglich lokaler Informationen und dem Austausch elektronischer Nachrichten insbesondere zwischen einander unmittelbar benachbarten Prozessoren ist die Vorgehensweise sehr robust gegenüber auftretenden Störungen und Ausfällen einzelner Prozessoreinheiten oder einzelner Verbindungen zwischen zwei Prozessoreinheiten.

[0025] Die Pixel-Anordnung, vorzugsweise die matrixförmige Pixel-Anordnung, wird in bestimmte Bereiche, beispielsweise in Bildblöcke partitioniert und in jedem Bereich wird eine Information verarbeitende Einheit, der Pixelprozessor, zugeordnet. Der Bereich kann einen Bildpunkt oder einen Sensor oder eine Mehrzahl von Bildpunkten bzw. Sensoren enthalten, die jeweils von einer Prozessoreinheit angesteuert werden. Vorzugsweise sind die Prozessoren in einem gröberen Raster als die Bildpunkte bzw. Sensoren selbst verteilt, was anschaulich einer räumlichen Unterabtastung entspricht. Auf diese Weise wird die Problematik der Verdrahtung und Adressierung über Spaltenleitungen und Zeilenleitungen bei einer matrixförmigen Pixel-Anordnung entschärft, da die jeweiligen Leitungen, welche die Prozessoreinheiten miteinander und mit einer Pixel-Anordnungs-Ansteuerungsschaltung verbinden, erfindungsgemäß großzügiger, das heißt räumlich dicker und somit mit einem geringeren elektrischen Widerstand behaftet, dimensioniert werden können. Jeder Pixelgruppenprozessor steuert den entsprechenden Bildbereich selbständig an, beispielsweise mittels einer Passiv-Matrix-Ansteuerung oder einer Aktiv-Matrix-Ansteuerung. Die Größe der Unter-Anzeigeeinheiten wird vorzugsweise derart gewählt, dass die oben beschriebenen Probleme der langsamen Aufladung der Pixelelemente sowie der sehr kur-

zen Zeitdauer zwischen zwei Ansteuerungszyklen bei einer gegebenen Auswahleinrichtung und einer gegebenen Verdrahtungstechnik nicht auftreten. In die Pixelgruppenprozessoren können im Prinzip beliebige Routingvorschriften für die darzustellende Bildinformation oder die zu erfassende Sensorinformation implementiert werden, so dass die Bildinformationen bzw. die Sensorinformationen in Form elektronischer Datenübertragung auch um auftretende Defekte im Display, das heißt in der Anzeigeeinrichtung, das heißt der Pixel-Anordnung (beispielsweise bei einer physikalischen Zerstörung wie einem punktuellen Defekt, bei Löchern, Rissen etc.), herumgeleitet werden können. Dies führt zu einer erhöhten Defekttoleranz der erfindungsgemäßen Vorrichtung. Erfindungsgemäß wird somit eine Pixel-Anordnung bereit gestellt sowie ein Verfahren bereit gestellt, mit dem es auf sehr einfache Weise möglich ist, den Abstand eines jeweiligen Pixels von einer Referenzposition und somit auch die lokale Position eines Pixels innerhalb der Pixel-Anordnung zu bestimmen. Die Bestimmung erfolgt nicht mehr basierend auf globalen Informationen, sondern auf lokalen Informationen, die jeweils in Form von Nachrichten zwischen zwei einander benachbarten Prozessoreinheiten ausgetauscht werden. Anders ausgedrückt wird das Problem der Abstandsbestimmung durch Selbstorganisation basierend auf lokalem Nachrichtenaustausch zwischen einander benachbarten Prozessoreinheiten gelöst. Das Selbstorganisationsverfahren weist somit unterschiedliche verteilte lokale uniforme Algorithmen auf, die die jeweiligen Nachrichten über die jeweilige bidirektionale Kommunikationsschnittstelle austauschen. Anders ausgedrückt bedeutet dies, dass erfindungsgemäß die Abstandsbestimmung auf einer Selbstorganisation auf der Basis rein lokaler Informationen erfolgt.

[0026] Unter einem Pixel ist erfindungsgemäß sowohl eine bildgebende Einheit, beispielsweise eine Flüssigkeits-Kristall-Bildschirmeinheit oder eine Polymerelektronik-Anzeigeeinheit oder allgemein jede Art von Bildschirm, welcher eine Vielzahl von Bildpunkten aufweist, zu verstehen. Alternativ kann die Erfindung in einer Pixel-Anordnung eingesetzt werden, in der zumindest ein Teil der Pixel als Sensorelemente ausgestaltet sind, das heißt die Pixel-Anordnung besteht in diesem Fall zumindest teilweise in einem Sensorfeld. Dies könnte beispielsweise auch eine Bildschirmeinheit mit in diese integriertem Touchpad sein, welche insbesondere auch auf mehrere Bildkacheln, anders ausgedrückt auf mehrere Bildbereiche aufgeteilt sein kann, wobei jedem Bildbereich eine Prozessoreinheit zugeordnet sein kann.

[0027] Bevorzugte Weiterbildungen der Erfindung ergeben sich aus den abhängigen Ansprüchen. Die im Weiteren beschriebenen Ausgestaltungen der Erfindung betreffen das erfindungsgemäße Verfahren sowie die erfindungsgemäße Prozessor-Anordnung.

[0028] Gemäß einer Weiterbildung der Erfindung ist jede Prozessoreinheit mit mindestens einem Pixel gekoppelt, so dass das Pixel von der jeweiligen ihm zugeordneten Prozessoreinheit steuerbar ist.

[0029] Gemäß dieser Ausgestaltung der Erfindung werden nicht nur die Abstände von miteinander gekoppelten Prozessoreinheiten, sondern darüber auch die Abstände der jeweiligen Pixel ermittelt. Dies ist eine wichtige Grundlage für ein im Rahmen der Darstellung von Bildinformation eingesetztes Routing eingehender darzustellender Bildinformation zu den einzelnen Pixeln, von denen die jeweilige Bildinformation dargestellt wird. In entsprechender Weise bildet die Abstandsbestimmung der Pixel auch eine Grundlage für ein Routing von erfasster Sensorinformation von dem Pixel zu den externen Schnittstellen zu beispielsweise dem Portalprozessor.

[0030] Gemäß einer anderen Weiterbildung der Erfindung ist zumindest ein Teil der Pixel jeweils als ein Sensor ausgestaltet. In diesem Fall wird erfindungsgemäß eine Abstandsbestimmung für einzelne Sensorpixel in einem großflächigen Sensorfeld durchgeführt.

[0031] Alternativ oder zusätzlich kann zumindest ein Teil der Pixel jeweils als ein bildgebendes Element ausgestaltet sein. In diesem Fall wird erfindungsgemäß eine Abstandsbestimmung für einzelne bildgebende Elemente in einem großflächigen, vorzugsweise matrixförmigen, Display durchgeführt.

[0032] Die Prozessoreinheiten können jeweils in einem hexagonalen Bereich angeordnet sein, in welchem Fall jede Prozessoreinheit jeweils sechs benachbarte Prozessoreinheiten aufweist, welche jeweils über eine bidirektionale Kommunikationsschnittstelle mit der Prozessoreinheit gekoppelt sind.

[0033] Ferner können die Pixel selbst eine hexagonale Form aufweisen.

[0034] Durch Einsatz einer hexagonalen Form wird eine sehr hohe Packungsdichte in der jeweiligen Anordnung erreicht.

[0035] Alternativ können die Prozessoreinheiten jeweils in einem rechteckigen Bereich angeordnet sein, in welchem Fall jede Prozessoreinheit jeweils vier benachbarte Prozessoreinheiten aufweist, welche jeweils über eine bidirektionale Kommunikationsschnittstelle mit der Prozessoreinheit gekoppelt sind.

[0036] Ferner können die Pixel selbst eine rechteckige Form aufweisen.

[0037] Gemäß einer anderen Ausgestaltung der Erfindung werden vor dem Bestimmen des Abstandes der Pixelprozessoren von der Referenzposition die örtlichen Positionen der Prozessoreinheiten innerhalb der Prozessor-Anordnung ermittelt, indem ausgehend von einer Prozessoreinheit an einer Einleitstelle der Prozessor-Anordnung jeweils Positionsermittlungs-Nachrichten, welche zumindest einen Zeilenparameter z und einen Spaltenparameter s aufweisen, welche die Zeilennummer bzw. Spaltennummer der die Nachricht sendenden Prozessoreinheit oder die Zeilennummer bzw. Spaltennummer der die Nachricht empfangenden Prozessoreinheit innerhalb der Prozessor-Anordnung enthält, an benachbarte Prozessoreinheiten übermittelt werden und von der jeweiligen Prozessoreinheit die folgenden Schritte durchgeführt werden:

- falls der Zeilenparameter in der empfangenen Nachricht größer ist als die bisher gespeicherte Zeilennummer der Prozessoreinheit, so wird der eigenen Zeilennummer der Prozessoreinheit der Zeilenparameterwert z der empfangenen Nachricht zugeordnet,
- falls der Spaltenparameter in der empfangenen Nachricht größer ist als die eigene Spaltennummer der Prozessoreinheit, so wird der gespeicherten Spaltennummer der Zeilenparameterwert der empfangenen Nachricht zugeordnet,
- falls die eigene Zeilennummer und/oder die eigene Spaltennummer aufgrund der oben dargestellten Verfahrensschritte verändert worden sind, so werden neue Positionsmess-Nachrichten mit neuen Zeilenparametern und neuen Spaltenparametern erzeugt, welche jeweils die Zeilennummer und Spaltennummer der die Nachricht sendenden

Prozessoreinheit oder die Zeilennummer und Spaltennummer der die Nachricht empfangenden Prozessoreinheit enthält, und diese werden über die bidirektionalen Kommunikationsschnittstellen an eine jeweilige Nachbar-Prozessoreinheit übertragen.

[0038] Durch diese Weiterbildung wird das erfindungsgemäße Konzept des lokalen Nachrichtenaustauschs zwischen einander benachbarten Prozessoreinheiten weiter ausgebaut, da schon die örtlichen Positionen der einzelnen Prozessoreinheiten innerhalb der Prozessor-Anordnung gemäß diesem Konzept basierend auf lokaler Positionsinformation, welche sich lediglich aus einer von der unmittelbar benachbarten Prozessoreinheit erhaltenen Positionsinformation ergibt, basiert. Dies ermöglicht eine sehr fehlerrobuste Vorgehensweise im Rahmen der Selbstorganisation der Prozessor-Anordnung.

[0039] Gemäß einer anderen Weiterbildung der Erfindung wird in einem iterativen Verfahren der eigene Abstandswert der Prozessoreinheit dann verändert wenn der bisher gespeicherte Abstandswert größer ist als der um einen vorgegebenen Wert erhöhte empfangene Abstandswert in der jeweils empfangenen Nachricht, und für den Fall, dass eine Prozessoreinheit den eigenen Abstandswert verändert, erzeugt diese eine Abstandsmess-Nachricht und sendet sie über alle Kommunikationsschnittstellen an benachbarte Prozessoreinheiten, wobei die Abstandsmess-Nachricht jeweils den eigenen Abstand als Abstandsinformation enthält oder den Abstandswert, den die empfangene Prozessoreinheit von dem Portalprozessor aufweist.

[0040] Der Abstandswert kann um einen um einen vorgegebenen Wert erhöhten Wert gegenüber dem eigenen Abstandswert verändert werden, vorzugsweise um den Wert "1".

[0041] Ausführungsbeispiele der Erfindung sind in den Figuren dargestellt und werden im Weitern näher erläutert. In den Figuren sind gleiche Komponenten mit identischen Bezugszeichen versehen.

[0042] Es zeigen

[0043] **Fig.** 1 eine Draufsicht auf eine Pixel-Anordnung gemäß einem ersten Ausführungsbeispiel der Erfindung;

[0044] **Fig.** 2a bis 2d Draufsichten sowie eine Schnittansicht von einzelnen Pixeln, die in der Pixel-Anordnung vorgesehen sein können, wobei das Pixel eine rechteckige Form aufweisen kann (**Fig.** 2a), eine dreieckige Form (**Fig.** 2b) oder eine hexagonale Form (**Fig.** 2c); **Fig.** 2d zeigt eine Schnittansicht durch die in den **Fig.** 2a bis 2c dargestellten Pixel;

[0045] **Fig.** 3 ein Blockdiagramm, in dem die Pixel-Anordnung mit Ansteuereinrichtung schematisch dargestellt ist;

[0046] **Fig.** 4 eine Draufsicht auf eine Prozessor-Anordnung gemäß dem ersten Ausführungsbeispiel der Erfindung;

[0047] **Fig.** 5 eine Draufsicht auf eine Prozessor-Anordnung gemäß einem zweiten Ausführungsbeispiel der Erfindung;

[0048] **Fig.** 6 eine Draufsicht auf eine Prozessoreinheit in hexagonaler Form;

[0049] **Fig.** 7a und 7b einen gerichteten Graphen (**Fig.** 7a) sowie einen ungerichteten Graphen (**Fig.** 7b);

[0050] **Fig.** 8 einen gerichteten Baum;

[0051] **Fig.** 9a und 9b eine Skizze einer Prozessor-Anordnung, modelliert als ungerichteter Graph (**Fig.** 9a) und als gerichteter Graph (**Fig.** 9b);

[0052] **Fig.** 10 eine Skizze unterschiedlicher Routing-Wege als gerichteter Baum mit einem Eingangsknoten als Wurzel;

[0053] **Fig.** 11 eine Skizze eines optimierten Routing-Baums;

[0054] **Fig.** 12a bis 12j eine Skizze des Routing-Baums aus **Fig.** 11 zu unterschiedlichen Ansteuerungszeitpunkten;

[0055] **Fig.** 13a bis 13f eine Skizze des Routing-Baums aus **Fig.** 11 zu unterschiedlichen Ansteuerungszeitpunkten;

[0056] **Fig.** 14 eine Draufsicht auf zwei hexagonale Prozessoreinheiten, in denen der bidirektionale Nachrichtenaustausch zwischen den zwei Prozessoreinheiten dargestellt ist;

[0057] **Fig.** 15 eine Skizze einer inkohärenten Prozessoreinheit;

[0058] **Fig.** 16 eine Skizze einer kohärenten Prozessoreinheit beim Versenden von MessKoherenz-Nachrichten;

[0059] **Fig.** 17 eine Skizze einer Prozessoreinheit, anhand der das Versenden von MessPosition-Nachrichten erläutert wird;

[0060] **Fig.** 18 eine Skizze einer Prozessor-Anordnung nach erfolgter Positionsbestimmung der einzelnen Prozessoreinheiten innerhalb der Prozessor-Anordnung;

[0061] **Fig.** 19 eine Skizze einer Prozessoreinheit, anhand der das Versenden einer MessDistance-Nachricht erläutert wird;

[0062] **Fig.** 20 die Prozessor-Anordnung nach erfolgter Abstandsbestimmung, wobei die Prozessor-Anordnung eine Vielzahl von Einleit-Prozessoreinheiten am unteren Rand der Prozessor-Anordnung aufweist;

[0063] **Fig.** 21 eine Prozessor-Anordnung nach erfolgter Abstandsbestimmung, wobei jeder dritten Prozessoreinheit in der untersten Zeile der Prozessor-Anordnung jeweils eine Referenzposition zugeordnet ist;

[0064] **Fig.** 22 eine Skizze einer Prozessoreinheit, anhand der das Empfangen und das Versenden von MessOrganize-Nachrichten erläutert wird;

[0065] **Fig.** 23 eine Skizze einer Prozessoreinheit, anhand der die Organisationsreihenfolge zum Versenden einer MessChannel-Nachricht in einer geradzahligen Spalte innerhalb der Prozessor-Anordnung dargestellt ist;

[0066] **Fig.** 24 eine Skizze einer Prozessoreinheit, anhand der die Organisationsreihenfolge zum Versenden einer MessChannel-Nachricht in einer ungeradzahligen Spalte innerhalb der Prozessor-Anordnung dargestellt ist;

[0067] **Fig.** 25 eine Skizze einer Mehrzahl von Prozessoreinheiten, anhand der die Organisation und der Nachrichtenaustausch über Kanäle, welche die Kommunikationsschnittstellen der Prozessoreinheiten miteinander koppeln, erläutert werden;

[0068] **Fig.** 26 eine Prozessor-Anordnung nach erfolgter regulärer Rückwärtsorganisation für den Fall, dass allen Prozessoreinheiten in der untersten Zeile der Prozessor-Anordnung Informationen von oder zu einem Portalprozessor zugeführt werden können oder gesendet werden können;

[0069] **Fig.** 27 eine Prozessor-Anordnung nach erfolgter regulärer Rückwärtsorganisation für den Fall, dass jeder dritten Prozessoreinheit in der untersten Zeile der Prozessor-Anordnung Informationen von oder zu einem Portalprozessor zugeführt werden können oder gesendet werden können;

[0070] **Fig.** 28 eine Skizze einer Prozessoreinheit, anhand der das Empfangen und Versenden von MessCountNodes-Nachrichten erläutert wird,

[0071] **Fig.** 29 eine Skizze einer Prozessoreinheit, anhand der das Empfangen und das Versenden von MessNodesSize-Nachrichten erläutert wird,

[0072] **Fig.** 30 die Prozessor-Anordnung nach erfolgter Ermittlung des Durchsatzes der Prozessoreinheiten für den Fall, dass allen Prozessoreinheiten in der untersten Zeile der Prozessor-Anordnung Informationen von oder zu einem Portalprozessor zugeführt werden können oder gesendet werden können;

[0073] **Fig.** 31 die Prozessor-Anordnung nach erfolgter Ermittlung des Durchsatzes der Prozessoreinheiten für den Fall, dass jeder dritten Prozessoreinheit in der untersten Zeile der Prozessor-Anordnung Informationen von oder zu einem Portalprozessor zugeführt werden können oder gesendet werden können;

[0074] **Fig.** 32 eine Skizze einer Prozessoreinheit, anhand der das Versenden von MessColDistance-Nachrichten erläutert wird;

[0075] **Fig.** 33 eine Skizze einer Prozessoreinheit, anhand der das Empfangen und das Versenden von MessBlockToken-Nachrichten erläutert wird;

[0076] **Fig.** 34 eine Skizze einer Prozessoreinheit, anhand der das Empfangen einer MessToken-Nachricht durch eine "ungefärbte" Prozessoreinheit dargestellt wird;

[0077] **Fig.** 35 die Prozessor-Anordnung nach erfolgter Ermittlung von Mäanderkanälen in der Prozessor-Anordnung bei erfolgter Tokenvergabe für den Fall, dass allen Prozessoreinheiten in der untersten Zeile der Prozessor-Anordnung Informationen von oder zu einem Portalprozessor zugeführt werden können oder gesendet werden können;

[0078] **Fig.** 36 eine Skizze einer Prozessoreinheit, anhand der das Empfangen und das Versenden von MessDeleteChannels-Nachrichten erläutert wird;

[0079] **Fig.** 37 eine Skizze einer Prozessoreinheit, anhand der das Empfangen und das Versenden von MessColOrganize-Nachrichten erläutert wird;

[0080] **Fig.** 38 die Prozessor-Anordnung nach erfolgter Reorganisation für den Fall, dass jeder dritten Prozessoreinheit in der untersten Zeile der Prozessor-Anordnung Informationen von oder zu einem Portalprozessor zugeführt werden können oder gesendet werden können;

[0081] **Fig.** 39 die Prozessor-Anordnung nach erfolgter Reorganisation für den Fall, dass allen Prozessoreinheiten in der untersten Zeile der Prozessor-Anordnung Informationen von oder zu einem Portalprozessor zugeführt werden können oder gesendet werden können;

[0082] **Fig.** 40 eine Skizze einer Prozessoreinheit, anhand der die Initialisierung der Einleit-Prozessoreinheits-Farbe mittels einer MessColDistance-Nachricht erläutert wird;

[0083] **Fig.** 41 die Prozessor-Anordnung nach erfolgter Reorganisation bei einem Gewicht $g = 0$ für den Fall, dass allen Prozessoreinheiten in der untersten Zeile der Prozessor-Anordnung Informationen von oder zu einem Portalprozessor zugeführt werden können oder gesendet werden können;

[0084] **Fig.** 42 die Pixel-Anordnung nach erfolgter Reorganisation bei einem Gewicht $g = \infty$ für den Fall, dass allen Prozessoreinheiten in der untersten Zeile der Prozessor-Anordnung Informationen von oder zu einem Portalprozessor zugeführt werden können oder gesendet werden können;

[0085] **Fig.** 43 eine Skizze einer Prozessoreinheit, anhand der das Empfangen und das Versenden von MessNumbering-Nachrichten erläutert wird;

[0086] **Fig.** 44 eine Skizze der Prozessor-Anordnung nach erfolgter Nummerierung für den Fall, dass allen Prozessoreinheiten in der untersten Zeile der Prozessor-Anordnung Informationen von oder zu einem Portalprozessor zugeführt werden können oder gesendet werden können;

[0087] **Fig.** 45 die Prozessor-Anordnung nach erfolgter Nummerierung für den Fall, dass jeder dritten Prozessoreinheit in der untersten Zeile der Prozessor-Anordnung Informationen von oder zu einem Portalprozessor zugeführt werden können oder gesendet werden können;

[0088] **Fig.** 46 eine Routing-Tabelle gemäß einem Ausführungsbeispiel der Erfindung;

[0089] **Fig.** 47 eine Skizze einer Prozessor-Anordnung, anhand der das Routing und die Darstellung von Pixeldaten erläutert wird;

[0090] **Fig.** 48 eine Skizze einer Prozessoreinheit, anhand der das Empfangen und Versenden von MessRetry-Nachrichten erläutert wird;

[0091] **Fig.** 49 eine Skizze einer Prozessor-Anordnung mit einer Vielzahl von Prozessoreinheiten und einer Vielzahl von Pixeln, wobei jeweils eine Gruppe von Pixeln einer Prozessoreinheit zugeordnet ist;

[0092] **Fig.** 50 eine Skizze einer 4×4-Pixelgruppe und deren Ansteuerung mittels einer Prozessoreinheit;

[0093] **Fig.** 51 eine Übersicht über die gemäß einem Ausführungsbeispiel der Erfindung verwendeten Nachrichten.

[0094] **Fig.** 1 zeigt eine Pixel-Anordnung **100** gemäß einem ersten Ausführungsbeispiel der Erfindung.

[0095] Die Prozessor-Anordnung **100** ist gebildet gemäß einem in [3] beschriebenen sogenannten "Fluidic Self Assembly" (FSA) der Firma Alien Technology™.

[0096] Die Prozessor-Anordnung **100** weist ein Substrat **101** mit einer Vielzahl von Vertiefungen **102** auf. Das Substrat **101** ist gemäß diesem Ausführungsbeispiel eine dünne, flexible Plastikfolie mit einer Vielzahl in diese gestanzten Vertiefungen, über welche eine Suspension mit einer Vielzahl von Prozessoreinheiten **103**, die als Computerchips **103** ausgestaltet sind, der Größe zwischen 50 µm bis 1000 µm, in einer Flüssigkeit geschwemmt werden. Die Computerchips **103** werden auch als Nanoblocks bezeichnet. Die Größe und Form der integrierten Computerchips **103**, welche im Weiteren als Prozessoreinheiten **103** bezeichnet werden, entspricht denen der Vertiefungen **102**, so dass sich die Prozessoreinheiten **103** in die Vertiefungen **102** selbsttätig, das heißt selbstordnend einfügen.

[0097] Dieser Vorgang ist schematisch in **Fig.** 1 dargestellt. Es entsteht auf die oben beschriebene Weise ein Feld bzw. ein Array von Prozessoreinheiten, die anschließend miteinander vermascht, das heißt elektrisch miteinander gekoppelt, werden.

[0098] Die Prozessor-Anordnung **100** wird zu einer Bildanzeigeeinrichtung ausgestaltet, bei der zur Ansteuerung der

Pixel die Prozessoreinheiten verwendet werden. Zu diesem Zweck wird über das gebildete Array von Prozessoreinheiten **103** ein elektro-optisches Medium (beispielsweise leuchtend, reflektierend, etc.), vorzugsweise ein Leuchtmedium gelegt, zu dem elektrische Kopplungen ausgehend von den entsprechenden das elektro-optische Medium steuernden Prozessoreinheiten **103** geschaffen werden.

[0099] Jede Prozessoreinheit **103** ist für das unmittelbar über ihm liegende Pixel zuständig, das heißt steuert dieses an, bzw. für die über ihm liegenden Pixelregion, das heißt ein Pixelbereich mit einer Mehrzahl von Pixeln, die zu einem Pixelbereich gruppiert sind.

[0100] Die Prozessoreinheiten **103** werden aus diesem Grunde im Weiteren auch als Pixelprozessoren **103** bezeichnet.

[0101] Es ist in diesem Zusammenhang anzumerken, dass die Erfindung keineswegs auf bildgebende Elemente als Prozessor-Anordnung bzw. Pixel-Anordnung beschränkt ist, sondern ebenfalls für ein Sensorarray mit einer Vielzahl von miteinander elektrisch gekoppelten und über jeweils eine oder mehrere Prozessoreinheiten **103** angesteuerten Sensoren einsetzbar ist.

[0102] Es können erfindungsgemäß auch Arrays verwendet werden, die teilweise bildgebende Elemente und teilweise Sensorelemente enthalten.

[0103] Die Form der einzelnen Prozessoreinheiten **103** können erfindungsgemäß beliebig ausgestaltet sein, beispielsweise wie gemäß **Fig.** 2a rechteckförmig, wie gemäß **Fig.** 2b dreieckförmig, oder wie gemäß **Fig.** 2c hexagonal.

[0104] **Fig.** 2d zeigt einen trapezoidalen Querschnitt durch die jeweiligen Prozessoreinheiten **103**. Durch den trapezoidalen Querschnitt wird gewährleistet, dass sich die einzelnen Pixel einfacher und mit größerer Wahrscheinlichkeit in den Vertiefungen **102** der Pixel-Anordnung selbstordnend einfügen.

[0105] Im Weiteren wird ohne Einschränkung der Allgemeingültigkeit eine Pixel-Anordnung, welche als bildanzeigendes, das heißt anders ausgedrückt als bildgebendes System ausgestaltet ist, beschrieben.

[0106] Das bildanzeigende System **300** gemäß einem Ausführungsbeispiel der Erfindung ist in **Fig.** 3 in einem Blockdiagramm schematisch dargestellt.

[0107] Das bildgebende System **300** weist eine Quelle **301** auf, welche die Quelle der anzuzeigenden Information darstellt und gemäß diesem Ausführungsbeispiel eine zentrale Verarbeitungseinheit (Central Processing Unit, CPU) einen Graphikprozessor, Sensoren oder sonstige Eingabegeräte und wahlweise weitere Komponenten enthält.

[0108] Weiterhin ist ein mit der Quelle **301** elektrisch gekoppeltes Anzeigeeinheits-Portal **302** vorgesehen, welches die Funktion eines Mittlers zwischen der Quelle **301** und der ebenfalls mit dem Anzeigeeinheits-Portal **302** gekoppelten eigentlichen Pixel-Anordnung **303** (welche die Prozessor-Anordnung und die Pixel aufweist) darstellt. Das Anzeigeeinheits-Portal **302** dient erfindungsgemäß zum Erzeugen eines Triggers, das heißt eines Anstoß-Signals, gemäß diesem Ausführungsbeispiel einer Anstoß-Nachricht, zur Initialisierung der Selbstorganisation, sowie der Informationsverteilung über Einleit-Prozessoreinheiten in der Prozessor-Anordnung **303**.

[0109] Die Prozessor-Anordnung **303** weist ein Feld aus uniformen, sogenannten intelligenten Pixeln oder Pixelregionen auf, die von Prozessoreinheiten **103** und den mit den Prozessoreinheiten **103** gekoppelten und von den Prozessoreinheiten **103** angesteuerten Pixeln gebildet werden. Die Prozessoreinheiten **103** sind untereinander vermascht, das heißt untereinander über jeweils den Prozessoreinheiten **103** zugeordnete bidirektionale Kommunikationsschnittstellen gekoppelt.

[0110] Einige Prozessoreinheiten **103** der Vielzahl von Prozessoreinheiten **103** in der Pixel-Anordnung **303** dienen als Einleitknoten zum Zuführen von Information aus dem Anzeigeeinheits-Portal **302** in die Prozessor-Anordnung **100** der Pixel-Anordnung **303**.

[0111] Gemäß diesem Ausführungsbeispiel hat der jeweilige Prozessor – im Weiteren bezeichnet als Portalprozessor – des Anzeigeeinheits-Portals **302** keinerlei Informationen über Größe und Ausgestaltung der Pixel-Anordnung **303**. Auch haben die einzelnen Prozessoreinheiten **103** oder die mit den Prozessoreinheiten **103** gekoppelten Pixel zu Beginn des Verfahrens keinerlei Informationen über ihre jeweilige Orientierung, d. h. Ausrichtung, oder ihre örtliche Position innerhalb der Prozessor-Anordnung.

[0112] Auch die einzelnen Pixelprozessoren, das heißt die Prozessoreinheiten, haben keinerlei Information über ihre eigene Ausrichtung und Lage, das heißt ihre lokale Position innerhalb der Prozessor-Anordnung.

[0113] In einer im Weiteren im Detail erläuterten Initialisierungsphase (vor dem ersten Einsatz der Pixel-Anordnung **303** oder nach einem erfolgten Zurücksetzen der gespeicherten Informationen in der Pixel-Anordnung **303**) stößt der Portalprozessor des Anzeigeeinheits-Portals **302** eine Selbstorganisation der Prozessor-Anordnung an, wie sie im Weiteren näher erläutert wird.

[0114] Im Rahmen der Selbstorganisation der Prozessor-Anordnung erlernen die Prozessoreinheiten **103** der Pixel-Anordnung **303** ihre Lage und Ausrichtung sowie Informationswege zum Bildaufbau, das heißt zum Zuführen darzustellender Bildinformation an die jeweiligen Pixel, welche die jeweilige Bildinformation tatsächlich darstellen sollen.

[0115] Dieser Lernprozess erfolgt unter Verwendung von Nachrichten, die zwischen jeweils einander benachbarten Prozessoreinheiten in der Pixel-Anordnung **303** ausgetauscht werden. Das erlernte Wissen wird teilweise wieder nach außen, das heißt an das Anzeigeeinheits-Portal **302** gegeben, und zwar in dem Maße, wie es das Anzeigeeinheits-Portal **302** später benötigt, um die Bildinformationen auf den richtigen Wegen und in der richtigen Abfolge der Pixel-Anordnung **303** zuzuführen zur Darstellung eines jeweils darzustellenden Bildes.

[0116] Für den Bildaufbau, das heißt für die Vorgehensweise bei der Informationsverteilung darzustellender Bildinformation innerhalb der Pixel-Anordnung **303** ist die Art der Bildinformation zu berücksichtigen.

[0117] Gemäß diesen Ausführungsbeispielen werden Einzelpixeldaten zur Darstellung eines Komplettbildes oder eines Differenzbildes, zum Beispiel im Rahmen von komprimierten Videobildern, zu den einzelnen Pixeln, das heißt genauer zu den einzelnen Pixelprozessoren **103** übertragen. Zu diesem Zweck wird jeder Pixelprozessor **103** von dem Portalprozessor des Anzeigeeinheits-Portals **302** individuell adressiert. Dies führt zu einem im Rahmen der Darstellung von Bildinformation erforderlichen Routings der Bildinformation zu den entsprechenden Pixeln und damit zu den entsprechenden Prozessoreinheiten innerhalb der Pixel-Anordnung. Im Rahmen des Routings von Einzelpixeldaten sind erfindungsgemäß folgende Besonderheiten des Routing-Problems zu berücksichtigen:

– Es werden Routing-Wege lediglich zwischen dem Portalprozessor des Anzeigeeinheits-Portals **302** und den einzelnen Pixelprozessoren, das heißt den Prozessoreinheiten der Pixel-Anordnung **303**, nicht aber zwischen den Pixelprozessoren untereinander bestimmt.
– Es liegt ein gleichmäßiges Routing-Aufkommen vor, das heißt pro anzuzeigendem digitalisierten Bild ist jedem Pixelprozessor genau ein Pixeldatum zu übermitteln.
– Es wird kein globales Wissen über die Gestalt des Netzes, das heißt der Vermaschung der einzelnen Pixelprozessoren innerhalb der Prozessor-Anordnung vorausgesetzt. Die Wahl von Routing-Wegen innerhalb der Prozessor-Anordnung erfolgt auf Basis lokaler Informationen, die zwischen den einzelnen Pixelprozessoren unter Verwendung von elektronischen Nachrichten ausgetauscht werden.

[0118] Somit sind erfindungsgemäß zwei Phasen im Rahmen des Einsatzes einer erfindungsgemäßen Pixel-Anordnung zu unterscheiden:
In einer ersten Phase, der sogenannten Selbstorganisation, wird durchgeführt eine

– Selbsterkennung der lokalen Positionen der einzelnen Pixelprozessoren innerhalb der Pixel-Anordnung und somit der Gesamtform der Pixel-Anordnung;
– Selbstorganisation von Routing-Wegen ausgehend von dem Portalprozessor, das heißt dem Prozessor des Anzeigeeinheits-Portals **302** zu jedem Pixelprozessor in der Pixel-Anordnung **303** derart, dass innerhalb einer vorgegebenen maximalen Zahl von Zeittakten jeder Pixelprozessor von dem Prozessor des Anzeigeeinheits-Portals **302** eine elektronische Nachricht zugeführt bekommen kann.

[0119] In einer zweiten Phase, dem eigentlichen Einsatz der Pixel-Anordnung **303** im Rahmen der Darstellung von Bilddaten, werden die Bilddaten von dem Portalprozessor zu den Pixelprozessoren versendet, das heißt übertragen, wodurch das darzustellende Bild in der Pixel-Anordnung **303** aufgebaut wird.
[0120] Die Pixelprozessoren **103** sind, wie in **Fig.** 4 dargestellt, für den Fall, dass sie eine rechteckige Form, vorzugsweise eine quadratische Form, aufweisen, jeweils über jede Seite des Vierecks über eine der somit jeweils vier vorgesehenen bidirektionalen Kommunikationsschnittstellen **401** pro Pixelprozessor **103** und darüber über elektrische Leitungen **402** jeweils mit dem unmittelbar zu einem jeweiligen Pixelprozessor **103** benachbarten Pixelprozessor **103** gekoppelt.
[0121] Anders ausgedrückt bedeutet dies, dass jeweils ein Nachrichtenaustausch zwischen zwei unmittelbar einander benachbarten Pixelprozessoren ermöglicht ist, nicht jedoch ein unmittelbarer, d. h. direkter Nachrichtenaustausch über eine weitere Entfernung hinweg als die unmittelbare Nachbarschaft eines Pixelprozessors **103**.
[0122] **Fig.** 5 zeigt ein anderes Ausführungsbeispiel, bei dem jeder Pixelprozessor **103** eine hexagonale Form aufweist und pro Pixelprozessor **103** sechs bidirektionale Kommunikationsschnittstellen **501**, ebenfalls an jeder Seite, das heißt Seitenkante, des jeweiligen Pixelprozessors **103** vorgesehen sind. Dies bedeutet, dass gemäß diesem Ausführungsbeispiel jeder Pixelprozessor **103** sechs Nachbar-Pixelprozessoren **103** aufweist, mit denen der jeweilige Pixelprozessor **103** über eine bidirektionale Kommunikationsschnittstelle **501** und eine elektrische Leitung **502** für den Austausch elektronischer Nachrichten gekoppelt ist.
[0123] Im Weiteren wird zur einfacheren Darstellung der Erfindung lediglich der Fall der hexagonalen Form eines Pixelprozessors **103** beschrieben, jedoch ohne Einschränkung der Allgemeingültigkeit.
[0124] Die Pixel-Anordnung **100** weist somit zwei Arten von Einzelkomponenten auf:

– Pixelprozessoren **103**, denen jeweils bis zu sechs bidirektionale Kommunikationsschnittstellen **501** und elektrische Leitungen **502** zugeordnet sind, und
– bidirektionale Verbindungen, im Weiteren auch als bidirektionale Kommunikationsschnittstelle **501** und der jeweiligen Kommunikationsschnittstelle **501** zugeordnete elektronische Leitung **502**, die je zwei Pixelprozessoren **103** oder einen Pixelprozessor **103** und den Portalprozessor miteinander koppeln.

[0125] Der hexagonale Pixelprozessor **103** kann sechs unterschiedliche Ausrichtungen aufweisen, wie in **Fig.** 6 dargestellt.
[0126] Gemäß **Fig.** 6 sind die einzelnen Verbindungen, das heißt damit auch die einzelnen Kommunikationsschnittstellen **501** während der Selbstorganisationsphase, wie sie im Weiteren noch näher erläutert wird, bereits orientiert worden. Die Verbindungen sind gemäß diesem Ausführungsbeispiel durchnummeriert und zum besseren Verständnis mit Himmelsrichtungen identifiziert, wobei gemäß diesem Ausführungsbeispiel folgende Nomenklatur verwendet wird:

– Eine erste Ausrichtung 0 (Ost) (Bezugszeichen **600**), anders ausgedrückt eine Ausrichtung nach rechts,
– eine zweite Ausrichtung **1** (Nordost) (Bezugszeichen **601**), anders ausgedrückt eine Ausrichtung nach rechts oben,
– eine dritte Ausrichtung **2** (Nordwest) (Bezugszeichen **602**), anders ausgedrückt eine Ausrichtung nach links oben,
– eine vierte Ausrichtung **3** (West) (Bezugszeichen **603**), anders ausgedrückt eine Ausrichtung nach links,
– eine fünfte Ausrichtung **4** (Südwest) (Bezugszeichen **604**), anders ausgedrückt eine Ausrichtung nach links unten, sowie
– eine sechste Ausrichtung **5** (Südost) (Bezugszeichen **605**), anders ausgedrückt eine Ausrichtung nach rechts unten.

[0127] Gemäß diesem Ausführungsbeispiel wird vorausgesetzt, dass der Portalprozessor des Anzeigeeinheits-Portals **302** elektrische Kopplungen zu Pixelprozessoren **103** auf ausschließlich einer Seite der Pixel-Anordnung **100** aufweist.
[0128] Definitionsgemäß sei dies die untere Seite der Pixel-Anordnung **100**, das heißt anschaulich die Südseite, wobei

die Kopplungen ebenfalls definitionsgemäß über die Südwest-Seite, das heißt über die fünfte Ausrichtungsrichtung der jeweiligen Pixelprozessoren **103** laufen.

[0129] Es ist in diesem Zusammenhang anzumerken, dass sowohl die Positionierung als auch die Ausrichtung der einzelnen Einleitstellen von Information zu den Pixelprozessoren **103** in der Prozessor-Anordnung **100** als auch die Form und die Ausrichtung der einzelnen Pixelprozessoren **103** in der Prozessor-Anordnung **100** grundsätzlich beliebig sind.

[0130] In unterschiedlichen Ausführungsformen der Erfindung ist der Portalprozessor

- mit allen Pixelprozessoren **103** der untersten Zeile in Form einer Matrix, das heißt in Zeilen und Spalten, angeordneten Pixelprozessoren **103** der Prozessor-Anordnung **100** elektrisch gekoppelt, oder
- mit Pixelprozessoren **103** der untersten Zeile der Prozessor-Anordnung in einem vorgegebenen, regulären, das heißt periodischen Abstand, das heißt beispielsweise jedem dritten, fünften, zehnten, etc., Pixelprozessor **103** innerhalb der untersten Zeile der Prozessor-Anordnung.

[0131] Der Portalprozessor kennt zwar nach erfolgter Fertigung der Pixel-Anordnung **303** die Anzahl seiner Verbindungen zu den Pixelprozessoren **103**, anders ausgedrückt die Anzahl von Einleitstellen zum Zuführen von Information zu Pixelprozessoren **103** innerhalb der Pixel-Anordnung **303**, aber nicht notwendigerweise die Dimension und die Ausgestaltung der Pixel-Anordnung **303** und damit der Prozessor-Anordnung, das heißt die tatsächliche Form und Anordnung der Pixelprozessoren **103** innerhalb der Prozessor-Anordnung **100**.

[0132] Es ist in diesem Zusammenhang anzumerken, dass insbesondere eine Richtungsangabe, beispielsweise die Südseite, nicht notwendigerweise eine gerade Linie innerhalb der Prozessor-Anordnung **100** darstellen muss.

[0133] Bei den im weiteren erläuterten Teil-Verfahren ist lediglich vorzusehen, dass die einzelnen Verbindungen zwischen dem Portalprozessor und den Pixelprozessoren **103** immer an derselben Stelle erfolgen sollten, gemäß diesem Ausführungsbeispiel über die Südwestseite **604**.

[0134] Die einzelnen Pixelprozessoren **103** oder die Verbindungen, welche beide als Oberbegriff als Einzelkomponenten der Prozessor-Anordnung bezeichnet werden, können folgende Zustände annehmen:

1. Fehlerfrei

[0135] Die jeweilige Komponente der Pixel-Anordnung arbeitet ohne Einschränkungen.

2. Defekt

[0136] Die jeweilige Komponente der Pixel-Anordnung ist vollständig ausgefallen. Ist die Komponente eine Prozessoreinheit, so sind ebenfalls alle Verbindungen zu dieser Prozessoreinheit als defekt zu deklarieren.

3. Instabil

[0137] Die Komponente weist Teilausfälle auf, beispielsweise arbeitet eine Richtung einer bidirektionalen Verbindung zwischen der jeweiligen Prozessoreinheit nur zeitweise (das heißt sie weist einen Wackelkontakt auf oder arbeitet methodisch falsch, beispielsweise ein Prozessor, der eine falsche Nachricht versendet).

[0138] Im Weiteren wird aus Gründen der einfacheren Darstellung der Erfindung der dritte Zustand nicht betrachtet, das heißt eine Komponente sei im Folgenden entweder fehlerfrei oder defekt. Daher spielt es gemäß diesen Ausführungsbeispielen keine Rolle, ob eine Komponente aufgrund einer speziellen Form der Pixel-Anordnung nicht existiert (das heißt beispielsweise eine Anzeigeeinheits-Folie, die die Form eines Dreiecks aufweist), oder ob die jeweilige Komponente aufgrund eines Herstellungsfehlers oder aufgrund erfolgter Abnutzung defekt wurde.

[0139] Bei der im Weiteren noch näher erläuterten Informationsweitergabe, das heißt dem Versenden von elektronischen Nachrichten zwischen zwei Pixelprozessoren **103** innerhalb der Pixel-Anordnung **303** oder von dem Portalprozessor zu einem Pixelprozessor **103** an einer Einleitstelle der Pixel-Anordnung **303** wird im Folgenden eine Taktung des Gesamtsystems, das heißt der gesamten Pixel-Anordnung **303** betrachtet.

[0140] Jeder Pixelprozessor **103** in der Pixel-Anordnung **303** ist derart eingerichtet, dass er innerhalb eines Zeittaktes folgende Aktionen durchführen kann:

- Lesen von einer oder mehreren elektronischen Nachrichten, die an einer oder mehreren Verbindungen, das heißt über eine oder mehrere bidirektionale Kommunikationsschnittstellen des jeweiligen Pixelprozessors anliegen und die im zeitlich vorangegangenen Zeittakt von einem Nachbar-Pixelprozessor versendet wurden.
- Verarbeitung der empfangenen Nachricht.
- Gegebenenfalls Versenden von einer oder mehreren Nachrichten über eine oder mehrere Verbindungen und damit über eine oder mehrere bidirektionale Kommunikationsschnittstellen des Pixelprozessors **103**, die im zeitlich darauffolgenden, das heißt nächsten Zeittakt von einem Nachbar-Pixelprozessor empfangen werden können.

[0141] Innerhalb eines Zeittaktes kann somit eine elektronische Nachricht nur von einem Pixelprozessor **103** zu einem Nachbar-Pixelprozessor **103** übertragen werden.

[0142] Es ist jedoch in diesem Zusammenhang anzumerken, dass erfindungsgemäß keine globale, das heißt für die gesamte Prozessor-Anordnung **100** vorgesehene gemeinsame Taktung der Pixelprozessoren **103** existieren muss, diese jedoch zur einfacheren Darstellung der Erfindung im Weiteren angenommen wird.

[0143] Im Weiteren werden zum einfacheren Verständnis der erfindungsgemäßen Vorgehensweise Grundlagen über die mathematische Modellierung der Pixel-Anordnung erläutert.

[0144] Im Weiteren werden die Pixelprozessoren **103** und das Anzeigeeinheits-Portal **302** gemeinsam als gerichteter

Graph sowie die Routing-Wege als gerichteter Baum modelliert.

[0145] Die Wahl eines Routings ergibt sich somit als diskretes Optimierungsproblem.

Definition 1

Gerichteter Graph, ungerichteter Graph

(i) Gegeben sei eine Menge V und eine Menge E. Ist

$g : E \rightarrow V^2 = V \times V$

eine Abbildung mit den Komponenten

$g^- : E \rightarrow V$ und $g^+ : E \rightarrow V$,

d. h.

$g : E \rightarrow V^2$,

$e \rightarrow (g^-(e), g^+(e))$,

so heißt das Tupel

(V, E, g)

gerichteter Graph mit Eckenmenge (Knotenmenge) V, Kantenmenge E und Inzidenzabbildung g. $g^-(e)$ heißt die initiale Ecke der Kante $e \in E$ und $g^+(e)$ heißt die terminierende Ecke der Kante $e \in E$.

(ii) Gegeben sei eine Menge V und eine Menge M. Man betrachte die Äquivalenzrelation

$\alpha := \{((x, y), (y, x)) \in V^2 \times V^2;$ mit $x, y \in V\} \subseteq V^2 \times V^2$

mit den Äquivalenzklassen

$[x, y] := \{(x, y), (y, x)\}$, für alle $x, y \in V$.

Mit einer Abbildung

$u : M \rightarrow V^2/\alpha = \{[x, y]; x, y \in V\}$

heißt das Tupel

(v, M, u)

ungerichteter Graph mit Eckenmenge (Knotenmenge) V, Kantenmenge M und Inzidenzabbildung u.

[0146] **Fig.** 7a zeigt einen gerichteten Graphen **700** und **Fig.** 7b zeigt einen ungerichteten Graphen **701**.

Definition 2

Terminierte Kanten, initiierte Kanten

[0147] Sei (V, E, g) ein gerichteter Graph und $v \in V$. Dann sei $E_{term}(v)$ die Menge der durch v terminierten Kanten, d. h.

$E_{term}(V) := \{e \in E; g^+(e) = v\}$,

und $E_{init}(v)$ die Menge der durch v initiierten Kanten, d. h.

$E_{init}(v) := \{e \in E; g^-(e) = v\}$.

Definition 3

Weg in einem gerichteten Graphen

[0148] Sei (V, E, g) ein gerichteter Graph, $K \subseteq E$.

(i) Für $a, b \in V$ und $n \in N$ definiere

$$\Gamma_K^n(a, b) := \left\{ \begin{array}{l} (k_1, \ldots, k_n) \in K^n; a = g^-(k_1) g^+(k_n) = b, \\ g^+(k_i) = g^-(k_{i+1}) \text{ für } \quad i = 1, \ldots, n-1, \\ \left| \left\{ a, g^+(k_1), \ldots, g^+(k_n) \right\} \right| = n + 1 \end{array} \right\}$$

als die Menge aller Wege von a nach b der Länge n mit Kanten K ($\Gamma_K^n(a, b) = \{\}$, falls kein solcher Weg existiert).
(ii) Für a, b ∈ V definiere

$$\Gamma_K(a, b) := \bigcup_{n \in N} \Gamma_K^n(a, b)$$

als die Menge aller Wege von a nach b mit Kanten aus K.

Definition 4

Gerichteter Baum

[0149] Sei (V, E, g) ein gerichteter Graph, V ≠ 0. (V, E, g) heißt gerichteter Baum, falls es ein w ∈ V gibt so dass

$|\Gamma_E(w, v)| = 1$, für alle $v \in V \backslash \{w\}$

und für alle $K \subseteq E, K \neq E$

$|\Gamma_K(w, v)| = 0$, für mindestens ein $v \in V \backslash \langle w \rangle$.

[0150] Das heißt es gibt genau einen Weg von w zu jeder Ecke v ≠ w und die Kantenmenge ist nicht verkleinerbar. Die eindeutige Ecke w heißt die Wurzel des gerichteten Baumes.
[0151] Die zweite Bedingung in der obigen Definition **4** garantiert die Eindeutigkeit der Wurzel, die sonst nicht gegeben wäre und verhindert die Existenz "überflüssiger" Kanten in dem Baum.
[0152] **Fig.** 8 zeigt ein Beispiel eines gerichteten Baumes **800** als einen Teil des in **Fig.** 7a skizzierten gerichteten Graphen.

Lemma 5

Eigenschaften eines gerichteten Baumes

[0153] Sei (V, E, g) ein gerichteter Baum. Dann gilt für alle a, b ∈ V

$|\Gamma_E(a, b)| + |\Gamma_E(b, a)| \leq 1$.

Definition 6

Weglänge, Durchsatz

[0154] Sei (V, E, g) ein gerichteter Baum mit Wurzel w ∈ V. Definiere

(i) Für jedes v E V\{w}, sei $\gamma_E(v) \in \Gamma_E(w, v)$ der eindeutige Weg von w nach v, d. h.

$\Gamma_E(w, v) = \{\gamma_E(V)\}$.

(ii) Für jedes $v \in V \backslash \{w\}$ gibt es ein $n \in N$ mit

$\{\gamma_E(v)\} = |\Gamma_E(w, y) = \Gamma_E^n(w, y)$.

Definiere $|\gamma_E(v)| := n$ als Weglänge des Weges $\gamma_E(v)$.
(iii) Definiere für $|V| < \infty$ und alle $v \in V$

$d_E(v) := 1 + |\{z \in V; \Gamma_E(v, z) \neq \{\}\}| \in N$

als Durchsatz des Knoters v.

Definition 7

Ast

[0155] Sei (V, E, g) ein gerichteter Baum. Definiere für alle $v \in V$

$$V_E(v) := \{v\} \cup \{z \in V; \Gamma_E(v, z) \neq \{\}\}$$

als Ast des Knotens v.

[0156] Es gilt folgendes Lemma:

Lemma 8

Mächtigkeit des Astes

[0157] Sei (V, E, g) ein gerichteter Baum und $v \in V$. Dann gilt

$$d_E(v) = |V_E(v)|.$$

[0158] Das Gesamtnetzwerk des bildgebenden Systems **300** inklusive dem Portalprozessor wird im Folgenden als ein Graph dargestellt. Um zu modellieren, dass existierende Verbindungen zwischen zwei Knoten stets in zwei Richtungen durchlässig sind, was eine bidirektionale Kommunikation symbolisiert, wird zunächst ein ungerichteter Graph betrachtet. Zur Festlegung des Routings wird anschließend ein gleichwertiger gerichteter Graph abgeleitet.

Definition 9

Display-Graph

[0159] Sei (V, M, u) ein ungerichteter Graph mit

(i)
$2 \leq |V| < \infty, 1 \leq |M| < \infty,$

(ii)
u injektiv (d. h. keine Zweiecke)
(iii)
$u(E) \cap \{[x, x]; x \in V\} = \{\}$ (d. h. schlingenfrei)

(iv) $w \in V$ sei ein ausgezeichneter Knoten und heiße Portal(knoten).

[0160] Es sei (V, E , g) der gerichtete Graph, für den gilt: Zu jedem $m \in M$ betrachte neue Elemente m^- und m^+ derart, dass

$$E := \{m^-; m \in M\} \cup \{m^+; m \in M\}, |E| = 2|M|.$$

[0161] Wähle die Abbildung g derart, dass

$$u(m) = \{g(m^-), g(m^+)\}, \text{ für alle } m \in M.$$

[0162] Gilt zusätzlich

(v) $\Gamma_E(w, v) \neq$ für alle $v \in V/\{w\}$ (d. h. zusammenhängend),
so heiße (V, E, g) ein Anzeigeeinheits-Graph, im Weiteren auch bezeichnet als Display-Graph.

[0163] Ein entsprechender ungerichteter Graph **900** (vgl. **Fig.** 9a) und der dazu gleichwertige gerichtete Pixel-Anordnungs-Graph **901** (**Fig.** 9b) sind in den **Fig.** 9a und **Fig.** 9b exemplarisch dargestellt.

[0164] Gemäß diesem Ausführungsbeispiel wird ein hexagonales 4×4-Pixelfeld mit einem Defekt gewählt. Die obige Definition **9** ist allgemein gehalten. Die betrachteten Netzwerke besitzen weitere einschränkende Eigenschaften, die hier zunächst aber nur kurz erwähnt werden:

- Mit Ausnahme des Portal-Knotens **902** ist die Anzahl der Kanten, denen ein Knoten **903** als initiale (terminierende) Ecke angehören kann, durch eine Zahl $q \in N$ beschränkt. Bislang wurden $q = 4$ (Orthogonales Netzwerk) und $q = 6$ (hexagonales Netzwerk) betrachtet.
- Der gerichtete Graph **901** ist im Allgemeinen ein ebener Graph bzw. ein plättbarer Graph (es sind Erweiterungen denkbar, bei denen dies nur für den Sub-Graphen gilt, der den Portal-Knoten **902** nicht enthält, falls die Zuleitungen **904** nicht am Rand der Pixel-Anordnung **100** eingespeist werden).

[0165] Für die weiteren Erläuterungen ist es sinnvoll, neben dem Portal-Knoten **902** noch diejenigen Knoten **903** aus-

zuzeichnen, die eine direkte Verbindung zu dem Portal-Knoten **902** aufweisen. Wie oben beschrieben werden diese Knoten als Einleitknoten **903** bezeichnet, das heißt sie repräsentieren die Referenzpositionen, denen die Einleit-Pixelprozessoren der Pixel-Anordnung zugeordnet sind. Die Kanten von dem Portal-Knoten **902** zu den Einleitknoten **903** werden im Weiteren als Zuleitungen **904** und die Kanten **905** zwischen Pixelprozessoren als Netzwerkverbindungen bezeichnet.

Definition 10

Zuleitungen, Netzwerkverbindungen, Einleitknoten

[0166] Es sei (V, E, g) ein Display-Graph mit Portalknoten w. Die Menge der Zuleitungen ist dann definiert durch

$$E_{port} := \{e \in E; g^-(e) = w\}$$

und die Menge der Netzwerkverbindungen durch

$$E_{net} := \{e \in E; g^-(e) \neq w \wedge g^+(e) \neq w\}.$$

[0167] Die Menge der Einleitknoten ist definiert durch

$$V_{port} := g^+(E_{Port}).$$

[0168] Im Weiteren wird die Problemstellung betrachtet, dass jedem Knoten eines Pixel-Anordnung-Graphen vom Portal-Knoten aus innerhalb eines Zeitrahmens (innerhalb einer Refresh-Rate) eine elektronische Nachricht übermittelt werden soll.

[0169] Geschieht dies, wie es diese Problemstellung nahe legt, auf fest gewählten Wegen und kreuzen sich Wege, die sich getrennt haben, nicht erneut, so bedeutet dies, dass ein gerichteter Baum als Unter-Graph des Pixel-Anordnungs-Graphen gewählt werden soll. Dieser gerichtete Graph, der auch als Routing-Baum bezeichnet wird, legt dann die Wege des Informationsflusses eindeutig fest, nicht aber die Dynamik des Informationsflusses.

[0170] Der Routing-Baum ist nicht eindeutig; im Allgemeinen ist die Menge aller möglichen Bäume unüberschaubar groß.

Definition 11

Zulässige Baum-Menge, zulässige Kantenmenge

[0171] Bei (V, E, g) ein Display-Graph mit Portalknoten $w \in V$. Die Menge aller zulässigen gerichteten Bäume in (V, E, g) ist definiert als
$B := \{(V, K, gl_K)$; mit $K \subseteq E$ und (V, K, gl_K) ein gerichteter Baum mit Wurzel w$\}$.

[0172] Die Menge aller zulässigen Kantenmengen bezüglich (V, E, g) ist dann definiert als

$$K := \{K \subseteq E; (V, K, gl_K) \in B\}.$$

[0173] Ein beispielhafter zulässiger Baum **1000** ist in **Fig.** 10 dargestellt mit den entsprechenden Routing-Wegen mit dem Portal-Knoten **1001** als Wurzelknoten des gerichteten Baums **1000**.

[0174] Basierend auf Definition **10** werden folgende Begriffe eingeführt:

Definition 12

Zuleitungen, Netzwerkverbindungen

[0175] Es sei (V, E, g) ein Display-Graph mit Portalknoten w und sei $K \in \kappa$. Die Menge der Zuleitungen in K ist dann definiert durch

$$K_{port} := E_{Port} \cap K.$$

[0176] Die Menge der Netzwerkverbindungen durch

$$K_{net} := E_{net} \cap K.$$

[0177] Zur Bewertung von Bäumen werden im Folgenden eine Reihe von Kriterien aufgeführt:

Definition 13

Baumbewertungen

[0178] Sei (V, E, g) ein Display-Graph mit Portalknoten $w \in V$ und der Menge κ der zulässigen Kantenmengen.

(i)

[0179] Für alle $v \in V\backslash\{w\}$ definiert

$$l_{min}(v) := \min_{K \in \kappa}\{|\gamma_K(v)|\}$$

den Abstand des Knotens v von der Wurzel w im Display-Graphen.

(ii)

[0180] Für alle $K \in \kappa$ definiert

$$L(K) := \max_{v \in V\backslash\{w\}}\{|\gamma_K(v)|\}$$

den Maximalabstand im durch K festgelegten Baum (V, K, gl_K).

$$L_{min} := \min_{K \in \kappa}\{L(k)\}$$

ist dann der Maximalabstand im Display-Graphen.

(iii)

[0181] Für alle $K \in \kappa$ definiert

$$D(K) := \max_{v \in V\backslash\{w\}}\{d_K(v)\}$$

den Maximaldurchsatz im durch K festgelegten Baum (V, K, gl_K).

$$D_{min} := \min_{K \in \kappa}\{D(K)\}$$

ist dann der Maximaldurchsatz im Display-Graphen.

[0182] Zur Auswahl der "besten" Bäume bzw. Kantenmengen sind mindestens die folgenden Probleme betrachtbar:

(i) Menge der Bäume, deren Knoten jeweils minimalen Abstand zur Wurzel haben:

$O_1 := \{K \in \kappa;\ |\gamma K(V)| = l_{min}(v)$ für alle $v \in V\backslash\{w\}$,

(ii) Menge der Bäume, deren Maximalabstand minimal ist:

$O_2 := \{K \in \kappa;\ L(K) = L_{min}\}$,

(iii) Menge der Bäume, deren Maximaldurchsatz minimal ist:

$O_3 := \{K \in \kappa;\ D(K) = D_{min}\}$

[0183] Es ist leicht einsehbar, dass $O_1 \subset O_2$.

[0184] Falls gilt $O_2 \cap O_3 \neq$, so sind alle Bäume aus $O_2 \cap O_3$ Minimierer der Funktionen L und K und als Routing-Baum besonders geeignet.

[0185] Falls $O_2 \cap O_3 \neq \{\ \}$ nicht gilt, so sind relaxierte Problemstellungen notwendig.

(iv) Menge der Bäume, deren Maximalabstand höchstens um $a \in N_0$ größer als minimal ist:
$O_4^a := \{K \in \kappa;\ L(K) \leq L_{min} + a\ \}$

(v) Menge der Bäume, deren Maximaldurchsatz höchstens um $b \in N_0$ größer als minimal ist:

$O_5^b := \{K \in \kappa;\ D(K) \leq D_{min} + b\}$.

[0186] Für eine geeignete Wahl von $a, b \in N_0$ wird stets

$O_4^a \cap O_5^b \neq \{\}$ möglich sein.

[0187] Die Problemstellung lässt sich aber auch als multikriterielles kombinatorisches Optimierungsproblem mit zwei Zielfunktionen auffassen.

[0188] Für den Anzeigeeinheits-Graphen gemäß **Fig.** 9b ist der Routing-Baum **1000** gemäß **Fig.** 10 sicherlich nicht optimal und zwar nach keiner der obigen Kriterien. Der Baum **1100** gemäß **Fig.** 11 liegt dagegen sogar in O_1 geschnitten mit O_3.

[0189] Oben wurde erläutert, wie die Wege des Informationsflusses im Anzeigeeinheits-Netzwerk durch die Auswahl eines Routing-Baums aus einer zulässigen Baum-Menge festgelegt werden können. Um die Anzeigeeinheits-Knoten mit den zum Bildaufbau nötigen Informationen zu versorgen, wird jedem Knoten vom Portal-Knoten ausgehend eine elektronische Nachricht entlang dieser Wege übermittelt. Eine parallele Übermittlung aller elektronischen Nachrichten ist im Allgemeinen nicht möglich, da gewisse Kapazitäten, wie viele Nachrichten in einem Zeittakt über eine Kante übertragen werden können und wie viele Nachrichten in einem Knoten zwischengespeichert werden können (Queue), nicht überschritten werden dürfen. Eine zeitliche Abfolge (Dynamik) des Informationsflusses sollte daher bestimmt werden.

[0190] Bei im Folgenden (V, E, g) ein Anzeigeeinheits-Graph mit Portal-Knoten w. Es sei $r := |V| - 1$ und $V = \{v_0, v_1, \ldots, v_r\}$, $v_0 = w$. Ist zudem $K \in \kappa$ vorgegeben, so können gewisse "Gesamt"-Routing-Matrizen τ und anschließend gewisse "Einzel"-Routing-Matrizen σ^l, $l = 1, \ldots, r$, eingeführt werden.

[0191] In τ wird die Information enthalten sein, wie viele elektronische Nachrichten über die einzelnen Kanten aus K in den einzelnen Zeittakten zu übertragen sind. Dabei werden Bedingungen an τ so formuliert, dass die Kapazitäten eingehalten werden und am Ende in jedem Knoten eine elektronische Nachricht vorhanden ist. Zwischen verschiedenen Nachrichten (das heißt den Einzelpixeldaten) wird in τ noch nicht unterschieden. Wie ein Routing eines speziellen Einzelpixeldatums an das jeweilige Zielpixel erfolgt bzw. erfolgen kann, ist aus τ noch nicht unmittelbar ersichtlich. Doch können aus τ gewisse "Einzel"-Routing-Matrizen σ^l $l = 1, \ldots, r$, abgeleitet werden, die genau dieses Routing der Einzelpixeldaten zu den Zielpixeln v_l, $l = 1, \ldots r$, beschreiben. Die "Einzel"-Routing-Matrizen σ^l, $l = 1, \ldots r$, sind dabei nicht notwendigerweise eindeutig, doch wird die Bewertung des Routings anhand der Routing-Dauer im Wesentlichen nur von τ abhängen. Im Weiteren wird daher ein Routing als bereits durch τ gegeben betrachtet.

Definition 14

Routing-Abbildung, Routing-Matrix

[0192] Es sei $K = \{k_1, \ldots, k_r\} \in \kappa$ (beachte: $|K| = |V| - 1$). Es seien $c_{port}, c_{net}, q \in N$. Eine ($c_{port}$, c_{net}, q)-Routing-Abbildung oder -Matrix über den durch K bestimmten Baum $(V, K, g|_K)$ ist eine Matrix

$$\tau = \left(\tau_{ij}\right)_{\substack{i=1,\ldots,n \\ j=1,\ldots,r}} \in N_0^{n,r},\ n \in N,$$

mit folgenden Eigenschaften

(i) $\tau_{ij} \le C_{port}$ für alle $j \in \{1, \ldots, r\}$ mit $k_j \in K_{port}$ und alle $i \in \{1, \ldots, n\}$, sowie $\tau_{ij} \le c_{net}$ für alle $j \in \{1, \ldots, r\}$ mit $k_j \in K_{net}$ und alle $i \in \{1, \ldots, n\}$,

(ii) für alle $v \in V\backslash\{w\}$ und $1 \le m \le n$ gilt

$$\sum_{1 \le i \le m-1} \sum_{\substack{1 \le j \le r, \\ k_j \in K_{term}(v)}} \tau_{ij} - \sum_{1 \le i \le m} \sum_{\substack{1 \le j \le r, \\ k_j \in K_{init}(v)}} \tau_{ij} \ge 0,$$

(iii) für alle $v \in V\backslash\{w\}$ und $1 \le m \le n$ gilt

$$\sum_{1 \le i \le m} \sum_{\substack{1 \le j \le r, \\ k_j \in K_{term}(v)}} \tau_{ij} - \sum_{1 \le i \le m} \sum_{\substack{1 \le j \le r, \\ k_j \in K_{init}(v)}} \tau_{ij} \le q,$$

(iv) für alle $v \in V\backslash\{w\}$ gilt

$$\sum_{1 \le i \le n} \sum_{\substack{1 \le j \le r, \\ k_j \in K_{term}(v)}} \tau_{ij} - \sum_{1 \le i \le n} \sum_{\substack{1 \le j \le r, \\ k_j \in K_{init}(v)}} \tau_{ij} = 1.$$

c_{port} heißt Kapazität der Zuleitungen, c_{net} heißt Kapazität der Netzwerkverbindungen und q heißt maximale Queue-Länge.

$|\tau| := n$

heißt Routing-Dauer. Die Menge aller (c_{port}, c_{net}, q)-Routing-Matrizen über $(V, K, g|_K)$ werde mit

$\Re c_{port}, c_{net}, q^{(K)}$

bezeichnet.

[0193] Die Erweiterung gegenüber den zuvor betrachteten Routing-Bäumen besteht in erster Linie darin, dass in τ zusätzlich eine zeitliche Komponente enthalten ist.

[0194] Der Matrixeintrag τ_{ij}, $i \in \{1,\ldots n\}$ $j \in \{1,\ldots r\}$ besagt, dass im i-ten Zeittakt τ_{ij} Nachrichten über die Kante k_j übertragen werden.

[0195] Bedingung (i) gewährleistet die Einhaltung von vorgegebenen Zuleitungskapazitäten und Netzwerkkapazitäten.

[0196] Bedingung (ii) sorgt für die nötige Kausalität im Netz. Nachrichten können nur dann von einem Knoten aus weitergeleitet werden, wenn sie zuvor (das heißt mindestens einen Zeittakt früher) an diesen Knoten übermittelt wurden.

[0197] Bedingung (iii) berücksichtigt Speicherplatzbeschränkungen in den Knoten.

[0198] Nach Bedingung (iv) schließlich liegt nach n Zeiteinheiten genau eine Nachricht in dem Knoten.

[0199] Die Routing-Matrix gibt somit gemeinsam mit dem Routing-Baum ein Routing-Verfahren mit Angabe der zeitlichen Abfolge der einzelnen Schritte an, das das Netz gleichzeitig mit Nachrichten versorgt.

[0200] Es wird definiert:

Definition 15

Routing

[0201] Es seien c_{port}, c_{net} $q \in N$. Ein (c_{port} ,c_{net}, q)-Routing ist ein Tupel (K, τ) bestehend aus einer zulässigen Kantenlänge

$$K = \{k1, \ldots, k_r\} \in \kappa$$

und einer Routing-Matrix $\tau \in R_{c_{port}, c_{net}, q}(K)$. Die Menge aller Routings werde mit $R_{c_{port}, c_{net}, q}$ bezeichnet.

[0202] Wie sich nun das dynamische Routing für jeden einzelnen Knoten ergibt, sehen wir im Folgenden.

[0203] Bestimme dazu Matrizen $\sigma^l \in \{0, 1\}^{n,r}$, $l = 1, \ldots r$, nach folgendem Algorithmus:

$\tau^0 := \tau;$

für $l = 1, \ldots, r$:

{

$\sigma^l := 0^{n,r} \in \{0,1\}^{n,r};$

sei $(k_{p_1}, \ldots, k_{p_z})$, $z \in N$, der Weg von w nach v_l;

$i_{z+1} := n + 1;$

für $y := z, \ldots, 1$ absteigend:

{

$$i_y := \max\left\{i \in \{1, \ldots, i_{y+1} - 1\} : \tau^{l-1}_{i,p_y} > 0\right\};$$

$$\sigma^l_{i_y p_y} := 1;$$

}

$\tau^l := \tau^{l-1} - \sigma^l;$

}

[0204] Man zeigt leicht, dass der Algorithmus wohldefiniert ist, und dass $\tau^r = 0^{n,r}$ gilt. Folglich ist

$$\sum_{1 \le l \le r} \sigma^l = \tau.$$

[0205] Es ist

$$\sum_{1 \le i \le n} \sum_{\substack{1 \le j \le r, \\ k_j \in K_{term}(v_{\tilde{l}})}} \sigma^l_{ij} - \sum_{1 \le i \le n} \sum_{\substack{1 \le j \le r, \\ k_j \in K_{init}(v_{\tilde{l}})}} \sigma^l_{ij} = \delta_{l\tilde{l}}.$$

für alle für alle $l, \tilde{l} \in \{1, \ldots, r\}$. Ein Matrixeintrag $\sigma^l_{ij} = 1$ besagt, dass die Nachricht an v_l im i-ten Zeittakt über die Kante k_j weitergeleitet wird.

[0206] Als Beweisskizze zu obiger Wohldefiniertheit des Algorithmus werden zwei Lemmata aufgeführt:

Lemma 16

Wohldefiniertheit von σ^l

[0207] Es sei $l \in \{1, \ldots, r\}$. Erfüllt $\tau^{l-1} \in N_0^{n,r}$ die Bedingung (ii) aus Definition **14** für alle $v \in V\backslash\{w\}$ und Bedingung (iv) aus Definition **14** für $v := e_l$, so lässt sich σ^l gemäß dem Algorithmus wählen.

Lemma 17

Eigenschaften von τ^l

[0208] Es sei $l \in \{1, \ldots, r\}$. Erfüllt $\tau^{l-1} \in N_0^{n,r}$ die Voraussetzungen von Lemma **16** und wird σ^l gemäß obigem Algorithmus gewählt, so erfüllt auch τ^l die Voraussetzungen von Lemma **16**.

Definition 18

Routing-Matrix zu einzelnem Knoten

[0209] Es seien $c_{port}, c_{net}, q \in N$. Es sei $(K, \tau) \in R_{c_{port},c_{net},q}$ und es seien die Matrizen σ^l, $l = 1, \ldots, r$, gemäß obigen Algorithmus gewählt. Dann heißen die σ^l, $l = 1, \ldots, r$, Routing-Matrizen zu den Knoten v_l, $l = 1, \ldots, r$ bzgl. (K, τ).

[0210] Oft gehen wir bei der Konstruktion der Matrizen τ und σ^l, $l = 1, \ldots, r$ umgekehrt vor. Wir legen Matrizen σ^l, $l = 1, \ldots, r$, fest, indem wir angeben, in welcher zeitlichen Abfolge die Nachricht an v_l über den Weg $\gamma_K(v_l)$ weitergeleitet wird. τ ergibt sich dann aus

$$\tau := \sum_{1 \le l \le r} \sigma^l .$$

[0211] Die zeitliche Abfolge des Routings zu jedem einzelnen Knoten und damit die σ^l, $l = 1, \ldots, r$, sind dabei so zu wählen, dass die Kapazitäten von Kanten und Knoten nicht überschritten werden, d. h. dass τ die Punkte (i) und (iii) aus Definition **14** erfüllt.

[0212] Im Weiteren werden Kriterien zu einem "günstigen" und nach Möglichkeit "optimalen" Auswahl von Routing-Verfahren in einem Anzeigeeinheits-Graphen angegeben. Es wird ein Routing im Weiteren dann als optimal bezeichnet, wenn es die kürzest mögliche Dauer beansprucht. Um dies in mathematische Sprache fassen zu können, werden folgende Begriffe eingeführt.

[0213] Bei dabei (V, E, g) stets ein Anzeigeeinheits-Graph und sei wie zuvor $V = \{v_0, \ldots V_r\}$ mit $v_0 = w$.

Definition 19

Minimale Routing-Dauer

(i) Bei $K = \{k_1, \ldots k_r\} \in \kappa$ und seien $c_{port}, c_{net}, q \in N$. Dann definiert

$$T^{min}_{c_{port},c_{net},q}(K) := \min_{\tau \in R_{c_{port},c_{net},q}(K)} \{\|\tau\|\}$$

die minimale Routing-Dauer über den durch K festgelegten Baum $(V, K, g|_K)$.

(ii) Seien $c_{port}, c_{net}, q \in N$. Dann definiert

$$T^{min}_{c_{port},c_{net},q} := \min_{K \in \kappa} \{T_{c_{port},c_{net},q}(K)\}$$

die minimale Routing-Dauer im Display-Graphen.

Definition 20

Optimales Routing

(i) Bei $K = \{k_1, \ldots k_r\} \in \kappa$ und seien $c_{port}, c_{net}, q \in N$. Unter einer optimalen Routing-Matrix im durch K festgelegten Baum $(V, K, g|_K)$ wird eine Routing-Matrix verstanden aus folgender Menge

$$R^{min}_{c_{port},c_{net},q}(K) := \left\{\tau \in R_{c_{port},c_{net},q}(K); |\tau| = T^{min}_{c_{port},c_{net},q}(K)\right\} .$$

(ii) Seien c_{port}, c_{net}, $q \in N$. Unter einem optimalen Routing wird ein Routing verstanden aus folgender Menge

$$R^{min}_{c_{port},c_{net},q} := \left\{\begin{matrix}(K, \tau); K = \{k_1, \ldots, k_r\} \in \kappa, \tau \in R_{c_{port},c_{net},q}(K) \\ \text{und} |\tau| = T^{min}_{c_{port},c_{net},q}\end{matrix}\right\}$$

[0214] Die Wahl einer optimalen Routing-Matrix bei bereits festgelegten Routing-Baum im Sinne von Definition **20** (i) ist einfach. Sie wird im vorliegenden Abschnitt für die Sonderfälle cport und $c_{net} = 1$ und c_{port} und $c_{net} > 1$ erläutert.

[0215] Die Lösung des in Definition **20** (ii) gestellten Optimierungsproblems bei freier Wahl des Routing-Baumes ist erheblich schwieriger. Um es exakt zu lösen, ist das Problem meist zu komplex. Im Folgenden werden aus diesem Grund heuristische Verfahren zu seiner Lösung erläutert. Die Lösung des Optimierungsproblems aus Definition **20** (i) bei festgelegtem Routing-Baum liefert dabei wichtige Strategien für die günstige Wahl des Routing-Baums.

[0216] Zunächst wird der Sonderfall erläutert, bei dem gilt $c_{port} = c_{net} = 1$.

[0217] Es sei $q \in N$ beliebig und $K \in \kappa$. Ohne Beschränkung der Allgemeingültigkeit gelte $K_{port} = E_{port}$ (ansonsten betrachte $u \in V_{port}^+(K_{port})$ nicht als Einleitknoten, setze also $V_{port} := g^+(K^{port})$).

[0218] Wegen $c_{port} = 1$ überlegt man leicht, dass

$$T^{min}_{c_{port},c_{net},q}(K) \geq \max_{v \in V_{Port}} d_K(v) = D(K) .$$

[0219] Es liegt sogar Gleichheit vor. Bei dazu

$$n := \max_{v \in V_{Port}} d_K(v) = D(K) .$$

[0220] Die Idee des folgenden Routings ist, dass in jedem Zeittakt über jede Zuleitung eine elektronische Nachricht in die Einleitknoten gelangt und in den folgenden Zeitintervallen schrittweise an ihren jeweiligen Zielknoten, das heißt den Ziel-Pixelprozessor, weitergeleitet wird. Es werden zunächst die Nachrichten an die weiter entfernten Knoten, später die Nachrichten an die nah am Portal-Knoten liegenden Knoten, das heißt Pixelprozessor, eingespeist. Ein entsprechendes Routing ist in den **Fig.** 12a bis **Fig.** 12i für den Fall $c_{port} = c_{net} = 1$ dargestellt. Mit den kleinen Vierecken ist jeweils eine elektronische Nachricht **1201** symbolisiert, welche über den Portal-Knoten **1202** zu den Einleit-Pixelprozessoren **1203** in die Pixel-Anordnung **100** geführt wird.

[0221] Es wird betrachtet $u \in V_{port}$ und es wird gesetzt

$d := d_K(u) = |V_K(u)$
Es sei

$$V_K(u) = \{v_{q_1}, \ldots v_{q_d}\} \text{ mit } v_{q_1} = u$$

derart angeordnet, dass

$$\Gamma_K\left(v_{q_i}, v_{q_j}\right) = \{\} \qquad (1)$$

für $i > j$. Dies ist insbesondere dann erfüllt, wenn

$$\left|\gamma_K\left(v_{q_i}\right)\right| \geq \left|\gamma_K\left(v_{q_j}\right)\right|$$

für $i > j$. Bei nun $l \in \{1, \ldots, d\}$ beliebig und sei

$$\left(k_{p_1}, \ldots, k_{p_z}\right),$$

$z \in N$, der Weg von w nach v_{q_l}.

[0222] Dann setze für alle $i \in \{1, \ldots, n\}$ und $j \in \{1, \ldots r\}$

$$\sigma^{q_l}_{ij} := \left\{\begin{matrix}1 \text{ falls } 1 + (d-1) \leq i \leq z + (d-1) \text{ und } p_{i-(d-1)} = j, \\ 0 \text{ sonst.}\end{matrix}\right\}$$

[0223] Um zu zeigen, dass σ^{q_l} eine Routing-Matrix zu v_{q_l} definiert, genügt es zu zeigen, dass

$z + (d - l) \leq n$,

denn dann reichen die n Zeittakte, um die Nachricht gemäß unserer Konstruktion von σ^{q_l} an ihr Ziel v_{q_l} zu leiten. Wegen (1) ist $l \geq z$ und damit

$z + (d - l) \leq d \leq n$

und damit ist es gezeigt.

[0224] Entsprechend obiger Überlegungen lassen sich durch Betrachtung aller Einleitknoten schließlich die σ^l für alle $l \in \{1, \ldots, r\}$ bestimmen. Wie üblich wird gebildet

$$\tau := \sum_{l=1}^{r} \sigma^l.$$

[0225] Man sieht leicht, dass τ dann wirklich ein (1, 1, q)-Routing über (V, K, g|K) für beliebige $q \in N$ definiert und nach obigen Überlegungen optimal ist. Es gilt also

$$T^{min}_{c_{port}, c_{net}, q}(K) = \max_{v \in V_{port}} d_K(v) = D(K).$$

[0226] **Fig.** 12a zeigt den Ausgangszustand, zu dem alle Nachrichten **1201** in dem Portal-Knoten **1202** gespeichert sind. Nach einem ersten Zeittakt sind die ersten zwei Nachrichten **1201** den Einleit-Pixelprozessoren **1203**, das heißt den Pixelprozessoren der Pixel-Anordnung **100**, über welche die Information über die Pixel-Anordnung zu den jeweiligen Pixelprozessoren zugeführt werden können, zugeführt und dort zwischengespeichert (vgl. **Fig.** 12b). Nach einem weiteren Zeitschritt (vgl. **Fig.** 12c) sind die ersten beiden Nachrichten schon an erste innere Knoten **1204** der Pixel-Anordnung übertragen und zwei weitere Nachrichten **1201** sind den Einleit-Pixelprozessoren **1203** zugeführt worden. Nach jeweils einem weiteren Zeitschritt ist die jeweilige elektronische Nachricht **1201** immer um jeweils einen Pixelprozessor weiter übertragen worden und es sind jeweils zwei neue Nachrichten **1201** in die Pixel-Anordnung **100** zugeführt worden, anders ausgedrückt den Einleit-Pixelprozessoren **1203** zugeführt. Die **Fig.** 12d, **Fig.** 12e, **Fig.** 12f, **Fig.** 12g, **Fig.** 12h, **Fig.** 12i zeigen das sukzessive Voranschreiten der Übertragung der Nachrichten bis zu ihrem jeweiligen Ziel-Pixelprozessor nach jeweils einem Zeittakt.

[0227] Als eine mögliche vorteilhafte Strategie für die Wahl eines optimalen Routings bei freier Wahl des Routing-Baums im Sinne von Definition **20** (ii) kann festgehalten werden:
Wähle den Routing-Baum so, dass alle Einleit-Knoten möglichst gleich großen (genauer: maximal um den Wert 1 unterschiedlichen) Durchsatz besitzen und setze die Routing-Matrix entsprechend obigen Überlegungen.

[0228] Im Weiteren wird der zweite Sonderfall kurz erläutert, bei dem gilt:

$c := c_{port} = c_{net} > 1, q \geq c.$

[0229] Bei $K \in \kappa$. Ohne Beschränkung der Allgemeingültigkeit gelte wiederum $K_{port} = E_{port}$.

[0230] In diesem Fall ist es schwieriger, die minimale Routing-Dauer im Voraus anzugeben. Es wird somit eine Routing-Matrix entwickelt, die ein optimales (c_{port}, c_{net}, q)-Routing über (V, K, g|K) definiert. Aus ihr kann schließlich die minimale Routing-Dauer ermittelt werden. Die Idee für diese Variante des Routings ist gleich der für den Fall $c_{port} = c_{net} = 1$ zuvor entwickelten, nur dass in diesem Fall stets $c = c_{port} = c_{net}$ Nachrichten gleichzeitig in einen Einleit-Knoten eingeleitet werden, um von dort aus an die am weitesten entfernten, noch nicht benachrichtigten Knoten weitergeleitet zu werden. Ein solches Routing ist wiederum in den **Fig.** 13a bis **Fig.** 13f skizziert.

[0231] Es sei zunächst

$$\tilde{n} := \max_{v \in V_{port}} d_K(v).$$

[0232] Es sei $u \in V_{port}$ und es sei $d := d_K(u) = |V_K(u)|$. Es sei

$$V_K(u) = (v_{q_1}, \ldots, v_{q_d}) \text{ mit } v_{q_1} = u$$

so angeordnet, dass

$$|\gamma_K(v_{q_i})| \geq |\gamma_K(v_{q_j})|$$

falls $i > j$. Sei

$$l \in \{1, \ldots, d\} \text{ und } \hat{d} := \left\lfloor \frac{d - l}{c} \right\rfloor,$$

d. h. die nächst kleinere ganze Zahl zu

$$\frac{d-1}{c}.$$

Sei

$$(k_{p_1}, \ldots, k_{p_z})$$

der Weg von w nach v_{q_l}. Setze nun für alle $i \in \{1..., \tilde{n}\}$ und $j \in \{1, \ldots, r\}$

$$\tilde{\sigma}_{ij}^{q_l} := \begin{cases} 1 & \text{falls } 1+\hat{d} \le i \le z+\hat{d} \text{ und } p_{i-\hat{d}} = j \\ 0 & \text{sonst.} \end{cases}.$$

[0233] Wie zuvor, bestimme auf die Weise $\tilde{\sigma}^l$ für alle $l \in \{1, \ldots, r\}$ und setze

$$\tilde{\tau} := \sum_{l=1}^{r} \tilde{\sigma}^l.$$

[0234] Streiche nun all diejenigen Zeilen in $\tilde{\tau}$, die gleich 0 sind, d. h. setze

$$n := \min\{\hat{n} \in N; \tau_{ij} = 0 \text{ für alle } \hat{n} < i \le \tilde{n} \text{ und } j = 1, \ldots, r\}$$

und

$$\tau := (\tilde{\tau}_{ij})_{\substack{i=1,\ldots,n \\ j=1,\ldots,r}}.$$

[0235] Man kann zeigen, dass τ ein optimales (c_{port}, c_{net}, q)-Routing über (V, K, glK) für beliebige $q \ge c$ definiert. Des Weiteren gilt

$$\left\lceil \frac{D(K)}{c} \right\rceil = \max_{v \in V_{port}} \left\lceil \frac{d_K(v)}{c} \right\rceil \le n \le \max_{v \in V_{port}} d_K(v) = D(K)$$

und

$L(K) \le n$.

[0236] Wie groß n nun tatsächlich ist, hängt von der konkreten Struktur der Äste der Einleitknoten ab, kann aber leicht berechnet werden. Dazu wird zunächst für jedes $u \in V_{port}$ die Anzahl der Zeittakte n_u berechnet, die benötigt wird, um alle Nachrichten an die Knoten des Astes von u zu routen. $V_K(u)$ und d seien dabei wie oben. Dann gilt:

$$n_u = \max_{l \in \{1, \ldots, d\}} \left(\left| \gamma_K(v_{q_l}) \right| + \left\lfloor \frac{d-1}{c} \right\rfloor \right).$$

[0237] Daraus ergibt sich die Routing-Dauer n als

$$n = \max_{u \in V_{port}} n_u.$$

[0238] Als alternative Strategie für die Wahl eines optimalen Routings bei freier Wahl des Routing-Baums im Sinne von Definition **20** (ii) ergibt sich somit:
Wähle den Routing-Baum so, dass alle Einleit-Knoten möglichst gleich großen Durchsatz besitzen und der Baum in den Ästen der Einleit-Knoten "ausreichend weit verzweigt" ist, so dass n möglichst nahe an

$$\left\lceil \frac{D(K)}{c} \right\rceil$$

herankommt. Setze die Routing-Matrix entsprechend obigen Überlegungen.
[0239] Eine "ausreichend weite Verzweigung" liegt anschaulich dann vor, wenn für alle Einleit-Knoten Folgendes gilt: Betrachte den Ast des Einleit-Knotens, ordne die zugehörigen Knoten nach aufsteigender Weglänge. Dann sollen sich

die Weglängen der Knoten nur alle c Knoten um den Wert 1 erhöhen, das heißt c Knoten der Weglänge **2**, c Knoten der Weglänge **3**, ...

[0240] Bei geringen Kapazitäten der jeweiligen Knoten und der Zuleitungen ist es wichtiger, auf einen gleichmäßigen Durchsatz in dem Einleit-Knoten zu achten, da in diesem Fall üblicherweise der Durchsatz durch die Einleit-Knoten der ausschlaggebende Faktor für die Begrenzung der Routing-Dauer nach unten ist. Die Einleit-Knoten stellen in diesem Falle gewissermaßen eine Engstelle des Baumes dar. Bei höheren Kapazitäten ist es dagegen wichtiger, auf ausreichend viele Verzweigungen im Baum und damit kurze Weglängen zu achten. Hier sind es üblicherweise die Weglängen, die die Routing-Dauer nach unten begrenzen. Sehr hohe Kapazitäten sind dagegen gar nicht mehr sinnvoll, da das hexagonale Netz die Zahl der Verzweigungen limitiert und gewisse minimale Weglängen von der Topologie des Netzes, das heißt die Topologie der Vermaschung bzw. Verkopplung der Pixelprozessoren in der Prozessor-Anordnung vorgegeben sind. Im Weiteren werden Ausführungsbeispiele der Verfahren zur Selbstorganisation der Pixelprozessoren in der Pixel-Anordnung erläutert.

[0241] Gemäß den Ausführungsbeispielen wird folgende Situation vorausgesetzt:

- Der zentralen externen Einheit, das heißt dem Portalprozessor ist die Topologie des Netzwerkes, das heißt die Anordnung der Pixelprozessoren in der Prozessor-Anordnung unbekannt.
- Die Pixelprozessoren sind durch bidirektionale Verbindungen miteinander vermascht.
- Eine direkte Kommunikation erfolgt nur zwischen jeweils einander unmittelbar benachbarten Nachbar-Pixelprozessoren.
- Basis der Kommunikation ist der Austausch von elektronischen Nachrichten, wie sie beispielsweise in **Fig.** 14 dargestellt sind.
- Jeder Kontakt mit anderen Komponenten zur Selbstorganisation (Positionsbestimmung, Erstellung von Routing-Tabellen etc.) und zum Bildaufbau wird durch verschiedene Nachrichten abgewickelt. **Fig.** 14 zeigt einen ersten Pixelprozessor **1401** mit hexagonaler Form sowie einen zweiten Pixelprozessor **1402**, ebenfalls in hexagonaler Form. Der erste Pixelprozessor **1401** weist sechs bidirektionale Kommunikationsschnittstellen **1403** auf, was durch jeweils einen Doppelpfeil in **Fig.** 14 angedeutet ist. Auch die zweite Prozessoreinheit **1402** weist sechs bidirektionale Kommunikationsschnittstellen **1404** auf. Die erste Prozessoreinheit **1401** und die zweite Prozessoreinheit **1402** sind über eine Zuleitung **1405**, das heißt eine elektrisch leitende Verbindung, welche selbstverständlich auch als optische Kommunikationsverbindung ausgestaltet sein kann oder als Funkverbindung miteinander derart gekoppelt, dass sowohl eine erste Nachricht **1406** von der ersten Prozessoreinheit **1401** zu der zweiten Prozessoreinheit **1402** übermittelt werden kann als auch eine zweite Nachricht **1407** von der zweiten Prozessoreinheit **1402** zu der ersten Prozessoreinheit **1401**.

[0242] Gemäß den vorliegenden Ausführungsbeispielen sind in fehlerfreiem Zustand alle Pixelprozessoren **1401, 1402** miteinander voll vermascht über die entsprechenden Zuleitungen und die bidirektionalen Kommunikationsschnittstellen.

[0243] Die oben genannte Problemstellung wird durch Selbstorganisation basierend auf lokalem Nachrichtenaustausch zwischen zwei einander unmittelbar benachbarten Prozessoreinheiten **1401, 1402** gelöst.

[0244] Das Selbstorganisationsverfahren besteht somit aus verteilten uniformen Algorithmen, die diese elektronischen Nachrichten über deren Kommunikationsschnittstellen übertragen.

[0245] Im Laufe des Verfahrens erlernen die Prozessoreinheiten **1401, 1402** ihre Ausrichtung und ihre ebene Position innerhalb der Prozessor-Anordnung sowie den Abstand der jeweiligen Prozessoreinheit zu dem Portalprozessor, allgemein zu einer Referenzposition. Die Referenzposition kann auch die Position einer Prozessoreinheit, welche sich an der Einleitstelle der Pixel-Anordnung befindet, sein. In weiteren Schritten werden lokal Routing-Wege zwischen den einzelnen Prozessoreinheiten und dem Portalprozessor geprägt. Die Algorithmen zur Wahl der Routing-Wege sind dabei derart ausgelegt, dass bei einem gleichmäßigen Informationsfluss die Routing-Dauer möglichst minimal wird. Die Selbstorganisation legt auch den Algorithmus zur Verteilung der Information bei Einsatz der Pixel-Anordnung im Rahmen des Bildaufbaus, das heißt im Rahmen der Darstellung eines digitalisierten Bildes auf der Pixel-Anordnung, fest. Aufgrund der speziellen Konzeption des Verfahrens spielen die Form der Pixel-Anordnung und damit auch ausgefallene Einzelkomponenten keine Rolle, womit erfindungsgemäß eine hohe Fehlertoleranz erreicht wird.

[0246] Das gesamte Verfahren weist eine Vereinigung folgender Teil-Verfahren auf:

- Uniforme Teilalgorithmen zur Nachrichtenverarbeitung, die von den Pixelprozessoren ausgeführt werden,
- dem Steueralgorithmus des Portalprozessors,
- einem Nachrichtenkatalog, welcher die Schnittstelle der Teilalgorithmen darstellt.

[0247] Im Folgenden wird eine hexagonale Vermaschung der Pixelprozessoren innerhalb der Pixel-Anordnung **100** ohne Einschränkung der Allgemeingültigkeit angenommen.

[0248] Die Übertragung der Algorithmen auf den orthogonalen Fall oder andere ebene Vermaschungen ist aber erfindungsgemäß vollkommen analog zu dieser unten gegebenen Darstellung.

[0249] Gemäß einem Kommunikations-Schichtenmodell unterhalb der erfindungsgemäß benötigten Funktionen liegende Funktionen, beispielsweise Ping-Nachrichten, die Sicherung der Übertragung mittels Prüfsummen, Empfangsbestätigung, Neuanforderung defekter Nachrichten etc. werden im Weiteren nicht berücksichtigt. Diese können jedoch ohne Weiteres im Rahmen der Erfindung implementiert sein.

[0250] Generell gilt bei den im Weiteren beschriebenen Verfahrensschritten, dass jeder Pixelprozessor aufgrund empfangener Nachrichten für jeden seiner Nachbar-Pixelprozessoren einen Datensatz anlegt, der die gewonnenen Informationen in einem dem jeweiligen Prozessor zugeordneten Speicher speichert.

[0251] In einem ersten Teil-Verfahren lernen die Pixelprozessoren eine gleichmäßige Ausrichtung.

[0252] Da alle Verbindungen des Portalprozessors gemäß obiger Konvention mit der Südwest-Seite der entsprechen-

den Einleit-Pixelprozessoren an den Einleitstellen verknüpft sind, kann dies zur Erzeugung der Kohärenz verwendet werden.

[0253] Dazu werden MessKoherenz-Nachrichten versendet, die als Parameter die Anzahl der Verbindungen enthalten, die die Empfangsverbindung gegen den Uhrzeigersinn von der Ostrichtung, wie oben definiert, entfernt ist.

[0254] Jeder Pixelprozessor ist zur Initialisierung als inkohärent gesetzt.

[0255] Bei Empfang einer Mess-Koherenz-Nachricht **1501** (vgl. **Fig.** 15) werden von dem die MessKoherenz-Nachricht **1501** empfangenden Prozessoreinheit **1500** die folgenden Schritte durchgeführt:

1. Falls die Prozessoreinheit **1500** schon kohärent ist, wird die Verarbeitung beendet.
2. Die Ost-Richtung wird anhand des Nachrichtenparameters bestimmt und alle Verbindungsbezeichnungen/Verbindungsnummerierungen werden entsprechend ausgerichtet.
3. Die Prozessoreinheit **1500** wird als kohärent gesetzt.
4. Über alle Verbindungen werden MessKoherenz-Nachrichten **1601, 1602, 1603, 1604, 1605, 1606** von der Prozessoreinheit **1500** ausgesendet, deren Parameter jeweils so gewählt werden, dass die die jeweilige MessKoherenz-Nachricht **1601, 1602, 1603, 1604, 1605, 1606** empfangenden Prozessoreinheiten **103** sich auf obige Weise korrekt ausrichten können (vgl. **Fig.** 16).

[0256] Das Teil-Verfahren zur gleichmäßigen Ausrichtung wird dadurch in Gang gesetzt, dass der Portalprozessor über seine Verbindungen die MessKoherenz-Nachricht (**2**) mit dem Parameterwert **2** an die jeweiligen Einleit-Pixelprozessoren übermittelt. Das Teil-Verfahren terminiert, wenn die letzte Prozessoreinheit kohärent geworden ist.

[0257] Die Zahl der benötigten Zeittakte zur Durchführung des Prozesses entspricht der maximalen Distanz eines Pixelprozessors von den Portalprozessor. Bis zum "Ersterben" der letzten Nachrichtenkommunikation können eventuell noch ein bis zwei Zeittakte mehr benötigt werden.

[0258] In einem weiteren Teil-Verfahren ermitteln die Pixelprozessoren mittels Austauschs elektronischer Nachrichten untereinander automatisch ihre örtliche Position innerhalb der Pixel-Anordnung.

[0259] Da das hexagonale Feld der Pixelprozessoren innerhalb der Pixel-Anordnung aus jeweils versetzen Zeilen besteht, wird das Koordinatensystem gemäß diesem Ausführungsbeispiel so gewählt, dass die Spaltennummern in den Zeilen abwechselnd geradzahlig oder ungeradzahlig sind.

[0260] Es ist in diesem Zusammenhang darauf hinzuweisen, dass bei einem orthogonalen Aufbau der Pixel-Anordnung das Koordinatensystem sehr einfach kanonisch wählbar ist.

[0261] Auf die oben beschriebene Weise wird es bei dem hexagonalen Feld ermöglicht, dass ein Prozessor unabhängig von der Geometrie der Pixel-Anordnung aus seiner eigenen Position (i, j) mit Zeile i und Spalte j die Positionen seiner Nachbar-Pixelprozessoren ermitteln kann.

[0262] Die jeweiligen Positionen sind in **Fig.** 17 für die Prozessoreinheit **1500** dargestellt. Wie man in **Fig.** 17 sieht, ist als Konvention vereinbart, dass die Spaltennummern von West nach Ost (von links nach rechts) ansteigen und die Zeilennummern von Süd nach Nord (von unten nach oben) ansteigen.

[0263] Zur Positionsbestimmung werden gemäß diesem Ausführungsbeispiel MessPosition-Nachrichten **1701, 1702, 1703, 704, 1705, 1706** ausgetauscht, die zwei Parameter enthalten, nämlich die Zeilennummer und die Spaltennummer, die die MessPosition-Nachricht **1701, 1702, 1703, 1704, 1705, 1706** sendende Prozessoreinheit als von ihr angenommene Position der die jeweilige Nachricht **1701, 1702, 1703, 1704, 1705, 1706** empfangenden Prozessoreinheit berechnet hat.

[0264] Zur Initialisierung ist die Position jedes Pixelprozessors als (0, 0) definiert. Der Prozess der Positionsbestimmung beginnt bei jedem Pixelprozessor, sobald er kohärent geworden ist, wie oben erläutert wurde.

[0265] Über alle Verbindungen werden dann die MessPosition-Nachrichten **1701, 1702, 1703, 1704, 1705, 1706** versendet, wie in **Fig.** 17 dargestellt.

[0266] Bei Empfang einer Mess-Position-Nachricht **1701, 1702, 1703, 1704, 1705, 1706** mit Zeilenparameter z und Spaltenparameter s werden von der jeweiligen empfangenden Prozessoreinheit die folgenden Schritte durchgeführt:

1. Falls $z > i$, wobei i die eigene Zeilennummer darstellt, so wird $i = z$ gesetzt.
2. Falls $s > j$, wobei j die eigene Spaltennummer darstellt, so wird $j = s$ gesetzt.
3. Falls sich aufgrund von Schritt **1** oder Schritt **2** eine Änderung der eigenen Position (i, j) ergeben hat, so werden über alle Verbindungen MessPosition-Nachrichten **1701, 1702, 1703, 1704, 1705, 1706** versendet, wie in **Fig.** 17 dargestellt.

[0267] Das Teil-Verfahren wird beendet, wenn sich keine Positionsänderungen mehr ergeben.

[0268] **Fig.** 18 zeigt ein Beispiel für die Prozessor-Anordnung **1800** mit verschiedenen Defekten, welches nach der oben beschriebenen Vorgehensweise automatisch die Positionen der einzelnen Prozessoren bestimmt hat. Gemäß diesem Ausführungsbeispiel wurden sowohl ausgefallene, d. h. fehlerhafte Prozessoren als auch ausgefallene Verbindungen verwendet. Dieses Ausführungsbeispiel dient auch im Weiteren Verlauf dieser Beschreibung in zwei Varianten mit einer unterschiedlichen Anzahl von Einleit-Prozessoreinheiten zur Beschreibung der weiteren Teil-Verfahren.

[0269] Die Zahl der benötigten Zeittakte zur Durchführung des Prozesses ist nach oben begrenzt durch die maximale Distanz eines Pixelprozessors von einem anderen Pixelprozessor in der Prozessor-Anordnung. Bis zum "Ersterben" der letzten Nachrichtenkommunikation können noch ein bis zwei Zeittakte mehr benötigt werden. Üblicherweise ist jedoch das Teil-Verfahren in Abhängigkeit der Geometrie der Prozessor-Anordnung **1800** in der Regel sogar schneller durchführbar.

[0270] In diesem Zusammenhang ist anzumerken, dass bei der Darstellung von hexagonalen Pixelbildern oder orthogonalen Pixelbildern von dem Portalprozessor eine Abbildung auf das so ermittelte Koordinatensystem der Prozessor-Anordnung **1800** durchgeführt wird. Bei dem in späteren Teil-Verfahren erfolgten Aufbau von Routing-Wegen werden

dem Portalprozessor die jetzt lokal gespeicherten Informationen übermittelt, so dass eine entsprechende Abbildung in dem Portalprozessor erfolgen kann.

[0271] In **Fig.** 18 ist in der Pixel-Anordnung jeweils für jeden Pixelprozessor **1801** die örtliche Position des jeweiligen Pixelprozessors **1801** innerhalb der Prozessor-Anordnung **1800** in Form eines Wertetuppels aufgetragen.

[0272] In einem zusätzlichen Teil-Verfahren wird der jeweilige Abstand einer Prozessoreinheit von dem Portalprozessor, das heißt die Länge des Weges von dem Pixelprozessor zum Portalprozessor (siehe auch Definition **6**) ermittelt, allgemein der Abstand einer Prozessoreinheit **1801** in der Prozessor-Anordnung **1800** von einer vorgegebenen Referenzposition.

[0273] Zur Initialisierung dieses Teil-Verfahrens ist der Abstand jeder Prozessoreinheit **1801** als "unendlich" definiert. Gemäß diesem Ausführungsbeispiel ist der Abstand jedes Pixelprozessors zu dem Portalprozessor als ein Wert definiert, der größer ist als ein maximaler Wert, der innerhalb der Pixel-Anordnung als Abstand angenommen werden kann.

[0274] Es wird ohne Einschränkung der Allgemeingültigkeit vorausgesetzt, dass die Schritte der oben dargestellten Teil-Verfahren bereits durchgeführt wurden.

[0275] Der Prozess der Abstandsbestimmung wird dann von dem Portalprozessor mittels Versendens von MessDistance(0)-Nachrichten an die Prozessoreinheiten an den Einleit-Stellen der Prozessor-Anordnung **1800** gestartet.

[0276] Bei Empfang einer MessDistance-Nachricht mit Abstandsparameter a werden von der jeweiligen die Mess-Distance-Nachricht empfangenden Prozessoreinheit die folgenden Schritte durchgeführt:

1. Falls $d \geq a + 1$, wobei d den eigenen Abstand darstellt, so wird $d = a + 1$ gesetzt.
2. Falls sich aufgrund von Schritt **1** eine Änderung des eigenen Abstandes d ergeben hat, so werden über alle Verbindungen MessDistance-Nachrichten **1901, 1902, 1903, 1904, 1905, 1906** an die jeweils benachbarten Prozessoreinheiten versendet (vgl. **Fig.** 19). Die jeweilige MessDistance-Nachricht **1901, 1902, 1903, 1904, 1905, 1906** enthält jeweils als Parameter den Abstandswert, den die Prozessoreinheit **1500** in dem vorangegangenen Schritt ermittelt hat.

[0277] Das Teil-Verfahren terminiert, wenn sich keine Abstandsänderungen mehr ergeben.

[0278] **Fig.** 20 und **Fig.** 21 zeigen die Prozessor-Anordnung **1800** gemäß einem ersten Ausführungsbeispiel und eine Prozessor-Anordnung **2100** gemäß einem zweiten Ausführungsbeispiel, wobei bei der Prozessor-Anordnung **1800** gemäß dem ersten Ausführungsbeispiel alle Prozessoreinheiten **2001** in der untersten Zeile **2002** der Prozessor-Anordnung **1800** mit dem Portalprozessor über deren Südwest-Seite **2003** gekoppelt sind.

[0279] Bei der Prozessor-Anordnung **2100** gemäß dem zweiten Ausführungsbeispiel enthält die unterste Zeile **2101** der Prozessor-Anordnung **2100** sowohl Prozessoreinheiten **2102**, welche nicht mit dem Portalprozessor gekoppelt sind als auch Prozessoreinheiten **2103**, welche über deren an der Südwest-Seite angeordneten Kommunikationsschnittstellen **2104** mit dem Portalprozessor gekoppelt sind. Gemäß dem zweiten Ausführungsbeispiel ist jede dritte Prozessoreinheit in der untersten Zeile **2101** über ihre an der Südwest-Seite liegenden Kommunikationsschnittstelle mit dem Portalprozessor verbunden.

[0280] Die Zahl der benötigten Zeittakte zur Durchführung des Prozesses entspricht der maximalen Distanz einer Prozessoreinheit von dem Portalprozessor. Wiederum können bis zum "Ersterben" der letzten Nachrichtenkommunikation noch ein bis zwei Zeittakte mehr benötigt werden.

[0281] In diesem Zusammenhang ist anzumerken, dass jede Prozessoreinheit aufgrund der jeweils empfangenen Nachrichten auch die Entfernung seiner direkten Nachbar-Prozessoreinheiten von dem Portalprozessor bei sich lokal zur späteren Verwendung speichern kann.

[0282] Anschaulich wird in diesem Teil-Verfahren in einem iterativen Verfahren der eigene Abstandswert der Prozessoreinheit dann verändert, wenn der bisher gespeicherte Abstandswert größer ist als der um einen vorgegebenen Wert erhöhte empfangene Abstandswert in der jeweils empfangenen Nachricht. Für den Fall, dass eine Prozessoreinheit den eigenen Abstandswert verändert, erzeugt dieser eine MessAbstands-Nachricht und sendet diese über alle Kommunikationsschnittstellen an benachbarte Prozessoreinheiten, wobei die MessAbstands-Nachricht jeweils den eigenen Abstand als Abstandsinformation enthält oder den Abstandswert, den die empfangene Prozessoreinheit von dem Portal-Prozessor aufweist, vorzugsweise einen um einen vorgegebenen Wert erhöhten Wert gegenüber dem eigenen Abstandswert, vorzugsweise einem Abstandswert, der um den Wert "1" erhöht ist.

[0283] Im Weiteren wird das Teil-Verfahren zur regulären Rückwärtsorganisation beschrieben.

[0284] Um die folgenden Verfahrensschritte durchführen zu können, ist es erforderlich, dass der Abstand eines Pixelprozessors zu einer jeweiligen Referenzposition ermittelt worden ist und somit bekannt ist und vorzugsweise als jeweilige Abstandsinformation in dem Speicher des jeweiligen Prozessors gespeichert ist.

[0285] In dem im Weiteren beschriebenen Teil-Verfahren werden die Verbindungen zwischen den jeweiligen Prozessoreinheiten im Weiteren als Kanäle bezeichnete Instanzen ausgezeichnet.

[0286] Die Mengen der Prozessoreinheiten mit dem Portalprozessor als Wurzel-Knoten und den Kanälen als Kanten zwischen den jeweiligen Prozessoreinheiten bilden einen Baum. Dieser Baum wird für das anschließende Routing verwendet, wie es oben in Zusammenhang mit den graphentheoretischen Grundlagen beschrieben worden ist.

[0287] Die Kanäle werden in regulärer Weise so bestimmt, dass jede Prozessoreinheit auf einem kürzesten Weg mit dem Portal-Knoten verbunden wird.

[0288] Zur Initialisierung ist jeder Pixelprozessor **1500** als "unorganisiert" definiert. Der Prozess der Organisation wird von dem Portalprozessor mittels Versendens von MessOrganize-Nachrichten **2201, 2202, 2203, 2204, 2205, 2206**, welche keinerlei Parameter aufweisen, über alle Verbindungen hinweg gestartet.

[0289] Bei Empfang einer MessOrganize-Nachricht **2201, 2202, 2203, 2204, 2205, 2206** werden von der jeweils die Nachricht empfangenden Prozessoreinheit die folgenden Schritte durchgeführt:

1. Falls die Prozessoreinheit schon organisiert ist, wird die Verarbeitung beendet.

2. Über alle Verbindungen mit Ausnahme der Empfangsverbindung, das heißt der Verbindung, über die die Mess-Organize-Nachricht **2201, 2202, 2203, 2204, 2205, 2206** empfangen worden ist, werden zusätzliche Mess-Organize-Nachrichten versendet (vgl. **Fig.** 22).

3. Die Prozessoreinheit ermittelt aufgrund der zuvor ermittelten Abstandsinformationen eine Nachbar-Prozessoreinheit, welche einen geringeren Abstand als sie selbst von der Referenzposition, vorzugsweise somit von dem Portalprozessor, aufweist. Es wird diejenige Nachbar-Prozessoreinheit ausgewählt und als "Vorgänger" definiert, welche als erste nach der gemäß **Fig.** 23 und **Fig.** 24 festgelegten Reihenfolge einen geringeren Abstand aufweist als die Prozessoreinheit selbst. Die Verbindung zwischen der Prozessoreinheit und ihrem "Vorgänger" wird besonders ausgezeichnet und "Kanal" genannt. Die Menge der Pixelprozessoren mit dem Portalprozessor als Knoten und den Kanälen als Kanten bilden dann einen Baum. Bei einem regulären Display ohne Fehler führt diese Vorgehensweise auf ein "Zickzack-Muster" bei der Festlegung der Kanäle.

4. Dem "Vorgänger" wird eine MessChannel-Nachricht geschickt und die Prozessoreinheit wird als organisiert gesetzt.

[0290] Bei Empfang einer MessChannel-Nachricht wird von dem die Mess-Channel-Nachricht empfangenden Prozessor der Absender als "Nachfolger" definiert. Entsprechend ist dann die Verbindung zwischen der Prozessoreinheit und dem "Nachfolger" ein Kanal.

[0291] Das Teil-Verfahren terminiert, nachdem alle Prozessoreinheiten auf diese Weise sich organisiert haben.

[0292] **Fig.** 25 zeigt beispielhaft eine organisierte Prozessoreinheit **2500**, wobei die Verbindungen **2501**, welche Kanäle sind, optisch hervorgehoben sind. Über die Kanäle **2501** werden beim Einsatz des Displays die Pixelinformationen, beispielsweise somit die darzustellenden Bildinformationen, geroutet.

[0293] **Fig.** 26 und **Fig.** 27 zeigen Beispiele für die Prozessor-Anordnung **1800** und **2100** nach erfolgter automatischer Organisierung, wie oben beschrieben.

[0294] Die Zahl der benötigten Zeittakte zur Durchführung des Teil-Verfahrens zur rückwärts gerichteten Selbstorganisation entspricht der maximalen Distanz eines Pixelprozessors vom Portalprozessor. Bis zum "Ersterben" der letzten Nachrichtenkommunikation können auch in diesem Fall ein bis zwei Zeittakte mehr benötigt werden.

[0295] Die reguläre Rückwärtsorganisation führt bei fehlerfreien rechteckigen Displays, das heißt Anzeigeeinheiten, zu gut balancierten Bäumen.

[0296] Da alle Prozessoreinheiten innerhalb der Prozessor-Anordnung **1800, 2100** auf jeweils einem kürzesten Weg mit dem Portal verbunden sind, bestimmt dieser Algorithmus ein Element der oben definierten "optimalen Menge" O_1. Im Falle von Horizontalrissen **2600, 2700**, wie in den **Fig.** 26 und **Fig.** 27 dargestellt, führt die oben beschriebene Vorgehensweise jedoch dazu, dass die durch den Riss verschatteten Anteile der Prozessor-Anordnung **1800, 2100** im Wesentlichen von einer einzigen Zuleitung vom Portal zum Display versorgt werden.

[0297] Daher werden im Weiteren noch zusätzliche alternative Möglichkeiten der Organisation beschrieben.

[0298] Zum Aufstellen von Routing-Tabellen ist der Durchsatz eines Pixelprozessors von erheblicher Bedeutung.

[0299] Der Durchsatz ist die Anzahl von Pixelinformationen, die zum Aufbau eines Bildes von diesem Prozessor jeweils verarbeitet oder weitergereicht werden müssen.

[0300] Die mathematische Definition des Durchsatzes ist in Definition 6 oben gegeben.

[0301] Diese Zahl ist identisch mit der Anzahl der Pixelinformationen, die über den Eingangs-Kanal empfangen werden.

[0302] Zum Durchführen der folgenden Teil-Verfahrensschritte muss in der Prozessor-Anordnung **1800, 2100** eine Baumstruktur beispielsweise mittels Kanäle organisiert worden sein, wie oben beschrieben.

[0303] Das Teil-Verfahren wird von dem Portalprozessor mittels Versendens von Mess-Count-Notes-Nachrichten, welche keine Parameter aufweisen, über alle Verbindungen zu den jeweiligen Einleit-Prozessoreinheiten gestartet.

[0304] Bei Empfang einer eingehenden MessCountNodes-Nachricht **2801** über den Eingangs-Kanal werden von dem jeweils die MessCountNodes-Nachricht empfangenden Prozessoreinheit die folgenden Schritte durchgeführt:

1. Über alle Ausgangs-Kanäle der die MessCountNodes-Nachricht empfangenden Prozessoreinheit werden wiederum MessCountNodes-Nachrichten **2802** versendet, wie in **Fig.** 28 dargestellt.

2. Alle Nachbar-Prozessoreinheiten, die über Ausgangs-Kanäle miteinander verbunden sind, werden mit einem Durchsatz mit dem Durchsatzwert "0" markiert.

3. Falls keine Ausgangs-Kanäle existieren, wird der eigene Durchsatz auf den Durchgangswert "1" gesetzt und es wird eine MessNodesSize-Nachricht **2901** über den Eingangs-Kanal an die jeweilige Vorgänger-Prozessoreinheit gesendet. **Fig.** 29 zeigt für eine Prozessoreinheit **1500** zwei eingehende MessNodesSize-Nachrichten, eine erste eingehende MessNodesSize-Nachricht **2901**, welche den Wert d_1 enthält und eine zweite eingehende MessNodesSize-Nachricht **2902** mit dem Parameter d_2. Bei Empfang einer MessNodesSize-Nachricht mit Durchsatzparameter $\hat{d}$ über einen Ausgangskanal werden von dem die MessNodesSize-Nachricht empfangenden Prozessoreinheit die folgenden Schritte durchgeführt:

1. Die Nachbar-Prozessoreinheit, von der die MessNodesSize-Nachricht **2901, 2902** empfangen wurde, wird mit dem Durchsatzparameter der MessNodesSize-Nachricht markiert.

2. Falls mindestens ein Ausgangs-Kanal mit einem Durchsatz mit dem Durchsatzwert "0" markiert ist, wird die Verarbeitung beendet.

3. Falls alle Ausgangs-Kanäle mit einem Durchsatzwert > 0 markiert sind, so wird der eigene Durchsatz d als Summe aller Ausgangs-Durchsätze +1 berechnet.

4. Es wird eine zusätzliche MessNodesSize-Nachricht **2903** von der Prozessoreinheit erzeugt und mit dem Durchsatzwert d, welche sich ergibt, gemäß folgender Vorschrift:
$d = d_1 + d_2 + 1$ gemäß dem oben beschriebenen Ausführungsbeispiel, über den jeweiligen Eingangs-Kanal gesendet.

[0305] Das Teil-Verfahren terminiert, nachdem der Portalprozessor über alle Verbindungen eine MessNodesSize-Nachricht erhalten hat.

[0306] Die Zahl der benötigten Zeittakte zur Durchführung des Teil-Verfahrens entspricht der doppelten maximalen Distanz eines Pixelprozessors vom Portalprozessor. Bis zum "Ersterben" der letzten Nachrichtenkommunikation können auch in diesem Fall noch ein bis zwei Takte mehr benötigt werden.

[0307] **Fig.** 30 und **Fig.** 31 zeigen Beispiele für die Prozessor-Anordnung **1800** bzw. **2100**, nachdem auf die oben beschriebenen Weise die Durchsätze automatisch ermittelt wurden.

[0308] In den jeweiligen Pixelprozessoren ist der jeweilige Durchsatzwert angegeben. Diese Beispiele zeigen, dass die Durchsätze derjenigen Einleit-Prozessoreinheiten sehr hoch sind, die das von dem jeweiligen Horizontalriss **2600, 2700** verschattete Gebiet der Prozessor-Anordnung **1800** bzw. **2100** versorgen müssen.

[0309] Daher wird im Weiteren ein alternatives Organisationsverfahren beschrieben, welches noch flexibler auf Fehler, das heißt Defekte und unreguläre Formen einer Prozessor-Anordnung **1800, 2100**, reagieren kann.

[0310] Um einen möglichst gleichmäßigen Durchsatz zu erreichen, besteht ein heuristischer Lösungsansatz zur Auswahl eines Routing-Baums im sukzessiven Versenden von sogenannten MessToken-Nachrichten, welche in der Prozessor-Anordnung **1800, 2100** "Plätze besetzen".

[0311] In Analogie zu einer allmählichen Einfärbung der Prozessor-Anordnung **1800, 2100** mittels Farbströmen wird jede Einleitstelle mit Token einer anderen "Farbe" beschickt. Auf diese Weise wird die Prozessor-Anordnung **1800, 2100** in Farbregionen unterteilt, die jeweils über eine Einleit-Prozessoreinheit von dem Portal-Knoten versorgt werden.

[0312] Anders ausgedrückt bedeutet dies, dass jeweils eine "Farbe" bzw. eine individuelle Markierung für jede von über eine jeweilige Einleit-Prozessoreinheit versorgte Prozessoreinheit vorgesehen ist.

[0313] Im Weiteren wird der Begriff "Farbe" zur anschaulicheren Darstellung und entsprechend ein mit gleicher Markierung markierter Bereich als "Farbregion" verwendet.

[0314] Es kommen folgende heuristische Strategien der Verteilung zum Tragen:

- Ein Tokengewicht bestimmt, um wie viel der Abstand zum Portal-Knoten maximal vergrößert werden darf aufgrund der Einfärbung.
- Einmal gefärbte Pixel, das heißt Prozessoreinheiten, bleiben gefärbt, anders ausgedrückt bleiben markiert.
- Die den Token versendende Prozessoreinheit wird zum "Vorgänger" und die Verbindung zu ihm zum Kanal. Im Weiteren nimmt das gefärbte Pixel, das heißt die markierte Prozessoreinheit, nur noch von dem jeweiligen Vorgänger Token an.
- Token werden bevorzugt über Kanäle versendet.

[0315] Nach der kompletten Einfärbung der Prozessor-Anordnung **1800, 2100** ist eine Reorganisation innerhalb der gefärbten Bereiche erforderlich, da sich aufgrund des Teil-Verfahrens nicht optimale "Mäander-Kanäle" **3501** bilden, wie beispielsweise in **Fig.** 35 dargestellt.

[0316] Zunächst werden in den folgenden Unterabschnitten die Teil-Verfahren zur Verarbeitung der bei der Tokenvergabe verwendeten Nachrichten beschrieben.

[0317] Die Abstandsbestimmung innerhalb einer Farbregion ist weitestgehend identisch zur allgemeinen, oben beschriebenen Abstandsbestimmung zu einer Referenzposition.

[0318] Der Farbabstand bestimmt dabei die Länge des kürzesten Weges einer Prozessoreinheit zum Portalprozessor, wobei alle Prozessoreinheiten des Weges derselben Farbregion angehören müssen.

[0319] Zur Initialisierung ist der Farbabstand jeder Prozessoreinheit als unendlich definiert und seine Farbe als undefiniert. Gemäß diesem Ausführungsbeispiel ist der Abstand jedes Pixelprozessors zu dem Portalprozessor als ein Wert definiert, der größer ist als ein maximaler Wert, der innerhalb der Pixel-Anordnung als Abstand angenommen werden kann. Ebenso markiert die Prozessoreinheit seine Nachbar-Prozessoreinheiten als undefiniert gefärbt mit Farbabstand unendlich.

[0320] Bei Empfang einer MessColDistance-Nachricht mit Farbe c und Farbabstandsparameter a werden von der jeweiligen die MessColDistance-Nachricht empfangenden Prozessoreinheit die folgenden Schritte durchgeführt:

1. Die die MessColDistance-Nachricht versendende Prozessoreinheit wird mit der Farbe c und dem Farbabstand a markiert.
2. Falls die Farbe c nicht mit der eigenen Farbe f übereinstimmt, das heißt der Farbe f der die MessColDistance-Nachricht empfangenden Prozessoreinheit, so wird die Verarbeitung beendet.
3. Der eigene Farbabstand d wird als Minimum der Farbabstände von gleichfarbig markierten Nachbarn plus den Wert 1 gesetzt.
4. Falls sich aufgrund von Schritt **3** eine Änderung des eigenen Farbabstandes d ergeben hat, so werden über alle Verbindungen MessColDistance-Nachrichten **3201, 3202, 3203, 3204, 3205, 3206** mit den Parametern (f, d), das heißt anders ausgedrückt mit dem eigenen Farbabstand d und der eigenen Farbe f versendet (vgl. **Fig.** 32).

[0321] Zum Blockieren von Nachbar-Prozessoreinheiten gegenüber empfangenen Token-Nachrichten werden erfindungsgemäß MessBlockToken-Nachrichten verwendet, das heißt nach Empfang einer solchen MessBlockToken-Nachricht dürfen zu diesen blockierten Nachbar-Prozessoreinheiten keine Token mehr versendet werden.

[0322] Gleichzeitig werden Farbe und Farbabstand wie bei der MessColDistance-Nachricht mitgeteilt.

[0323] Zur Initialisierung sind alle Nachbar-Prozessoreinheiten einer Prozessoreinheit als unblockiert gesetzt.

[0324] Bei Empfang einer eingehenden MessBlockToken-Nachricht **3301** mit der Farbe c und dem Farbabstandsparameter a als Nachrichtenparameter werden von der jeweils die MessBlockToken-Nachricht empfangenden Prozessoreinheit die folgenden Schritte durchgeführt:

1. Die die MessBlockToken-Nachricht versendende Prozessoreinheit wird als blockiert gesetzt und mit der Farbe c und dem Farbabstand a markiert.
2. Falls die Farbe c nicht mit der eigenen Farbe f, das heißt der Farbe des die Mess-Block-Token-Nachricht empfangenden Prozessoreinheit übereinstimmt, wird die Verarbeitung mit dem weiter beschriebenen Schritt **5** fortgesetzt.
3. Der eigene Farbabstand d wird als Minimum der Farbabstände von gleichfarbig markierten Nachbar-Prozessoreinheiten plus dem Wert 1 gesetzt.
4. Falls sich aufgrund von Schritt **3** eine Änderung des eigenen Farbabstands d ergeben hat, so werden von der Prozessoreinheit MessColDistance-Nachrichten **3201, 3202, 3203, 3204, 3205, 3206** über alle Verbindungen mit Parametern (f, d) versendet, wie in **Fig.** 32 dargestellt.
5. Falls es einen Eingangs-Kanal gibt und alle Nachbar-Prozessoreinheiten als blockiert gesetzt sind, so wird eine MessBlockToken-Nachricht **3302** mit den Parametern (f, d) erzeugt und über den Eingangs-Kanal versendet, wie in **Fig.** 33 dargestellt.

[0325] Zum Einfärben, das heißt zum Markieren von Prozessoreinheiten und somit zum Definieren von Farbregionen, das heißt zu markierten Bereichen innerhalb der Prozessor-Anordnung **1800, 2100** werden erfindungsgemäß sogenannte MessToken-Nachrichten verwendet.

[0326] Bei der Verarbeitung von Mess-Token-Nachrichten ist zu unterscheiden, ob die Prozessoreinheit noch ungefärbt oder schon von einem Token gefärbt wurden.

[0327] Bei Empfang einer eingehenden MessToken-Nachricht **3401** mit dem Gewicht g und der Farbe f als Nachrichtenparameter werden von einer ungefärbten Prozessoreinheit, welche die Mess-Token-Nachricht **3401** empfängt, die folgenden Schritte durchgeführt:

1. Der potentielle eigene Farbabstand pd wird als Minimum der Farbabstände von mit der Farbe f gefärbten Nachbar-Prozessoreinheiten +1 gesetzt.
2. Falls das Gewicht $g \leq pd - a$ ist, wobei a der Abstand (nicht der Farbabstand!) der Prozessoreinheit von dem Portalprozessor ist, so wird der die MessToken-Nachricht **3401** versendenden Prozessoreinheit eine MessBlock-Token-Nachricht geschickt und die Verarbeitung wird beendet (die Ausbreitung der Tokens wird daher durch einen relaxierten Abstand beschränkt).
3. Die die MessBlockToken-Nachricht **3401** sendende Prozessoreinheit wird als blockiert gesetzt. Die eigene Farbe wird als f gesetzt und der eigene Farbabstand als pd.
4. Der die Mess-Token-Nachricht **3401** sendenden Prozessoreinheit wird eine MessChannel-Nachricht geschickt und die Prozessoreinheit wird als organisiert gesetzt. Somit ist der Eingangs-Kanal festgelegt.
5. Über alle Verbindungen mit Ausnahme des Eingangs-Kanals der Prozessoreinheit **1500** werden MessBlockToken-Nachrichten **3402, 3403, 3404, 3405, 3406** versendet, wie in **Fig.** 34 dargestellt, um eine Tokenvergabe von dort zu verhindern.
6. Falls alle Nachbar-Prozessoreinheiten als blockiert gesetzt sind, so wird eine MessBlockToken-Nachricht **3402, 3403, 3404, 3405, 3406** über den Eingangs-Kanal gesendet, wie in **Fig.** 33 dargestellt.

[0328] Bei Empfang einer Mess-Token-Nachricht mit dem Gewicht g und der Farbe f über den Eingangs-Kanal wird hingegen von einer schon gefärbten Prozessoreinheit anders vorgegangen.

[0329] Man betrachtet bei einer geraden Spaltennummer eine Reihenfolge R = (SE, SW, E, W, NE, NW), was einer Reihenfolge R entspricht von (Südost, Südwest, Ost, West, Nordost, Nordwest) und bei einer ungeraden Spaltennummer eine Reihenfolge R = (SW, SE, W, E, NW, NE), was entspricht einer Reihenfolge (Südwest, Südost, West, Ost, Nordwest, Nordost) und führt die folgenden Verfahrensschritte durch:

1. Falls die empfangene Mess-Token-Nachricht nicht über den Eingangs-Kanal kam oder die Farbe f nicht mit der eigenen Farbe übereinstimmt, wird die Verarbeitung beendet.
2. Falls es nach der Reihenfolge R einen unblockierten Ausgangs-Kanal gibt, so wird über diesen Ausgangs-Kanal eine MessToken-Nachricht mit den Parametern (g, f) geschickt, das heißt, das Token wird weitergereicht, und die Verarbeitung wird beendet.
3. Falls es nach der Reihenfolge R eine unblockierte Verbindung gibt, so wird über diese Verbindung eine MessToken-Nachricht (g, f) versendet und die Verarbeitung wird beendet.
4. Über den Eingangs-Kanal wird eine MessBlockToken-Nachricht geschickt, da sich das Token nicht weiterreichen lässt.

[0330] Da bei der Wahl der Farbregionen die Kanäle aufgrund des oben beschriebenen Teil-Verfahrens nicht optimal gesetzt werden können, wie in **Fig.** 35 dargestellt, werden diese Kanäle mit MessDeleteChannels-Nachrichten gelöscht und später neu gesetzt. Zur Terminierung des Teil-Verfahrens wird die Nachricht mit einem Parameter "stamp" versehen, dessen Wert nicht identisch ist mit dem entsprechend gespeicherten Parameter in der Prozessoreinheit. In diesem Zusammenhang ist anzumerken, dass der Portalprozessor bei jeder Reorganisation einen anderen Parameter "stamp" verwendet.

[0331] Bei Empfang einer eingehenden MessDeleteChannels-Nachricht **3601** mit dem Parameter "stamp" werden von der die jeweilige MessDeleteChannels-Nachricht empfangenden Prozessoreinheit die folgenden Schritte durchgeführt:

1. Falls der eigene Stempelparameter identisch zu dem empfangenen Parameterwert "stamp" ist, wird die Verarbeitung beendet.
2. Der eigene Stempelparameter wird auf den Wert in der MessDeleteChannels-Nachricht "stamp" gesetzt.
3. Alle Kanäle werden gelöscht.

4. Über alle Verbindungen mit Ausnahme der Verbindung zu der die MessDeleteChannels-Nachricht sendenden Prozessoreinheit werden MessDeleteChannels-Nachrichten **3602, 3603, 3604, 3605, 3606** mit dem Parameter "stamp" gesendet, wie in **Fig.** 36 dargestellt.

[0332] Nach Löschen der alten Kanäle werden neue Kanäle innerhalb einer Farbregion mittels Verwendens von MessColOrganize-Nachrichten gesetzt.

[0333] Die Verarbeitung von eingehenden MessColOrganize-Nachrichten **3701** und das Versenden von MessColOrganize-Nachrichten **3702, 3703, 3704, 3705, 3706** ist weitestgehend identisch zur Verarbeitung von MessOrganize-Nachrichten, wie oben beschrieben.

[0334] Ein Unterschied besteht jedoch darin, dass die betrachteten Nachbar-Prozessoreinheiten identisch wie die verarbeitende Prozessoreinheit eingefärbt sein müssen und dass nicht der Abstand, sondern der Farbabstand als Kriterium verwendet wird.

[0335] Zur Durchführung des oben beschriebenen Teil-Verfahrens sollten im Pixel-Array alle beschriebenen Schritte bis zur Abstandsbestimmung wie oben erläutert durchgeführt worden sein.

[0336] Wie oben in dem ersten Ausführungsbeispiel werden die Verbindungen speziell als "Kanäle" ausgezeichnet.

[0337] In einem ersten Schritt wird von dem Portalprozessor über alle Verbindungen je eine MessColDistance-Nachricht **4001** (vgl. **Fig.** 40) mit den Parametern (f, 0) mit unterschiedlichem Farbparameter f versendet. Somit markieren alle Nachbar-Prozessoreinheiten das Portalprozessor dies mit einer unterschiedlichen Farbe.

[0338] Auf diese Weise ist gewährleistet, dass ausgehend von jeder Einleit-Prozessoreinheit jeweils eine individuelle und eindeutige Markierung erfolgt.

[0339] In einem zweiten Schritt werden von dem Portalprozessor über alle Verbindungen sukzessive MessToken-Nachrichten mit den Parametern (g, f) mit identischen Gewicht $g \in N_0$ und unterschiedlichem Farbparameter f versendet, um alle Prozessoreinheiten der Prozessor-Anordnung **1800, 2100** einzufärben.

[0340] Das Teil-Verfahren terminiert, wenn über alle Verbindungen des Pixelprozessors MessBlockToken-Nachrichten eingetroffen sind, das heißt wenn die Prozessor-Anordnung **1800, 2100** komplett eingefärbt wurde.

[0341] Es ist in diesem Zusammenhang anzumerken, dass die gesamte Prozessor-Anordnung **1800, 2100** mit diesem Verfahren immer komplett eingefärbt werden kann.

[0342] **Fig.** 38 zeigt die Prozessor-Anordnung **2100** für den Fall, dass sie mit dem Gewicht g = 4 eingefärbt wurde und bei der der Durchsatz nach der Organisation dargestellt wurde. Wie man im Vergleich mit **Fig.** 30, die mittels regulärer Rückwärtsorganisation gebildet wurde, sieht, ist der Baum erheblich besser balanciert.

[0343] Allerdings bilden sich aufgrund der Konstruktion dieses Teil-Verfahrens innerhalb der gefärbten Bereiche Mäander-Wege **3801**, so dass die Prozessoreinheiten nicht durch die kürzest mögliche Distanz mit dem Portalprozessor verbunden sind.

[0344] Daher wird in einem dritten Schritt vom Portalprozessor über alle Verbindungen eine MessDeleteChannels-Nachricht, wie oben erläutert, geschickt, um die gebildeten Kanäle zu löschen. Direkt nach dieser Nachricht wird über alle Verbindungen eine MessColOrganize-Nachricht geschickt, die innerhalb der gefärbten Bereiche neue Kanäle bildet, welche dann kürzeste Verbindungen darstellen.

[0345] Das Teil-Verfahren terminiert, nachdem sich alle Prozessoreinheiten auf diese Weise organisiert haben. Die Zahl der benötigten Zeittakte zur Durchführung der Prozesse entspricht dem maximalen Farbabstand eines Pixelprozessors vom Portalprozessor. Bis zum "Ersterben" der letzten Nachrichtenkommunikation können auch in diesem Fall noch ein bis zwei Takte mehr benötigt werden.

[0346] Der erzeugte Routing-Baum hängt von dem Gewicht g, welcher als Parameter in der jeweiligen Mess-Token-Nachricht enthalten ist, ab.

[0347] **Fig.** 39 zeigt die Prozessor-Anordnung **1800** für nach erfolgter Reorganisation mit Gewicht g = 4 und die entsprechenden Mäander-Wege **3901**.

[0348] Das Gewicht g gibt an, um wie viel der Farbabstand einer Prozessoreinheit größer sein darf als der Abstands selbst. Je größer das Gewicht g ist, desto besser balanciert wird üblicherweise der entstehende Baum sein, aber desto länger sind üblicherweise auch die Pfade in diesem Baum. Zur Erläuterung ist auf **Fig.** 41 hinzuweisen, in der die Prozessor-Anordnung **1800** nach erfolgter Bildung der Mäander-Wege mit dem Gewicht a = 0 und auf die **Fig.** 42 in der die Prozessor-Anordnung **1800** nach erfolgter Bildung der Mäander-Wege mit dem Gewicht $g = \infty$ gezeigt sind.

[0349] Die beste Wahl des Gewichts hängt üblicherweise von den Transporteigenschaften der jeweiligen Verbindungen ab, das heißt davon, wie viele Nachrichten pro Zeittakt über eine Verbindung versendet werden können. Je kleiner diese Zahl ist, desto größer wird üblicherweise da beste Gewicht sein müssen.

[0350] Zuvor wurden zwei Verfahren zur Auswahl eines Routing-Baums beschrieben.

[0351] Wenn ein Routing-Baum ausgewählt wurde, das heißt wenn die entsprechenden Kanäle ausgewählt wurden, so kann ein optimales Routing für diesen Baum auf sehr einfache Weise ermittelt werden. Die Grundlagen hierzu wurden im Rahmen der Beschreibung der graphentheoretischen Grundlagen erläutert.

[0352] In einem ersten Schritt werden alle Pixelprozessoren, das heißt die Prozessoreinheiten innerhalb der Prozessor-Anordnung **1800, 2100**, durchnummeriert.

[0353] Die Nummern werden anschließend beim Routing als Zieladressen verwendet. In einem zweiten Schritt werden die gesammelten lokalen Informationen von den jeweiligen Prozessoreinheiten dem Portalprozessor übermittelt. In dem Portalprozessor wird anschließend die Gesamt-Routingtabelle erstellt.

[0354] Gemäß diesem Ausführungsbeispiel werden MessNumbering-Nachrichten zur Durchnummerierung aller Prozessoreinheiten in der Pixelanordnung **1800, 2100** verwendet. Voraussetzung ist, dass der Durchsatz der jeweiligen Prozessoreinheiten bereits ermittelt wurde, beispielsweise gemäß dem oben beschriebenen Teil-Verfahren.

[0355] Das Teil-Verfahren der Nummerierung wird von dem Portalprozessor mittels Versendens von MessNumbering-Nachrichten **4301** über die Ausgangskanäle des Portalprozessors, welche den Einleit-Prozessoreinheiten übermittelt werden, gestartet.

[0356] Wenn für die entsprechenden Nachbarprozessoreinheiten Durchsätze d_1, d_2, d_3, ... ermittelt wurden, so wird der jeweilige MessNumbering-Nachricht **4302** der Parameter **1**, **1** + d_1, 1 + d_1 + d_2, ... als Nachrichtenparameter mit übertragen.

[0357] Nach Empfang einer MessNumbering-Nachricht **4301** mit dem Parameter n über den jeweiligen Eingangskanal der Prozessoreinheit (vgl. **Fig.** 43) werden von der die MessNumbering-Nachricht **4301** empfangenden Prozessoreinheit die folgenden Schritte durchgeführt:

1. Die eigene Nummer der Prozessoreinheit wird auf den Wert n, die dem Wert der empfangenen MessNumbering-Nachricht **4301** entspricht, gesetzt.
2. Über alle Ausgangskanäle der Prozessoreinheit wird je eine von der Prozessoreinheit erzeugte zusätzliche MessNumbering-Nachricht **4302** erzeugt und mit den Parametern n + 1, n + d_1 + 1, n + d_1 + d_2 + 1, ... versendet, wobei d_1, d_2, ... die Durchsätze der entsprechenden Nachbar-Prozessoreinheiten sind.

[0358] Das Teil-Verfahren terminiert, wenn der letzte Prozessor durch die letzte Prozessoreinheit durchnummeriert worden ist. Die Zahl der benötigten Zeittakte zur Durchführung des Teil-Verfahrens entspricht der maximalen Distanz einer Prozessoreinheit über Kanäle vom Portalprozessor. Bis zum "Ersterben" der letzten Nachrichtenkommunikation können auch bei diesem Teil-Verfahren noch ein bis zwei Zeittakte mehr benötigt werden.

[0359] Die **Fig.** 44 und **Fig.** 45 zeigen die Pixelanordnungen **1800** (**Fig.** 44) und **2100** (**Fig.** 45) nach erfolgter Numerierung der einzelnen Prozessoreinheiten innerhalb der jeweiligen Pixelanordnung.

[0360] Die Nummer einer Prozessoreinheit kann einfacherweise als Adresse zum Routing von Daten oder auch Bildern verwendet werden, da jedem Ausgangskanal eine Prozessoreinheit ein eindeutiges Nummernintervall zugeordnet ist. Jede Prozessoreinheit kann somit eine einfache Routing-Tabelle anlegen.

[0361] Im Beispiel von **Fig.** 45 lautet beispielsweise die Tabelle für die mit der Nummer **123** nummerierte Prozessoreinheit wie in der Routing-Tabelle **4600** in **Fig.** 46 dargestellt ist.

[0362] Die lokal erzeugten Informationen werden dem Portalprozessor mittels MessCollectInfo-Nachrichten mitgeteilt, die die folgenden Nachrichtenparameter enthalten:

- Die Position der jeweiligen Prozessoreinheit innerhalb der jeweiligen Pixelanordnung, das heißt die Zeile und die Spalte, in der sich die Prozessoreinheit befindet,
- die Pixelnummer,
- der Abstandswert, mit dem der Abstand der Prozessoreinheit von dem Portalprozessor angegeben wird,
- der Farbabstand, und
- Durchsatz der Prozessoreinheit.

[0363] Die MessCollectInfo-Nachrichten werden von den Prozessoreinheiten jeweils gesendet, sobald die jeweilige Prozessoreinheit durchnummeriert worden ist.

[0364] Mit diesen Informationen kann der Pixelprozessor zum einen Pixel-Bilder auf das reale Pixelfeld abbilden (sampling) und zum anderen diese Bilddaten mit Hilfe der Pixelnummern routen.

[0365] Beim Versenden eines Gesamtbildes, das heißt beim Zuführen der Daten an alle Prozessoreinheiten, werden dabei die Nachrichten zuerst versendet, die den längsten Weg haben, wie oben im Rahmen der Beschreibung der graphentheoretischen Grundlagen erläutert.

[0366] Aus dieser Routing-Tabelle ergibt sich dann auch unmittelbar die Routing-Dauer, mit der die Routing-Bäume bewertet werden.

[0367] Mit Hilfe der Pixelnummern und den vorab beschriebenen Routing-Tabellen kann ein Pixelbild beim weiteren Betrieb des Displays auf sehr einfache Weise versendet werden. Dazu verschickt der Portalprozessor Nachrichten vom Typ MessRGB die mit folgenden Parametern versehen sind:

- Die Nummer des Pixels, welches adressiert wird, und
- die Farbinformation für dieses Pixel, beispielsweise Rot-Grün-Blau-Werte.

[0368] **Fig.** 47 zeigt ein Beispiel für eine Bilddarstellung auf der Pixelanordnung. Selbstverständlich ist die Darstellung unabhängig vom gewählten Routing-Baum.

[0369] Zuvor wurde die Auswahl und das Bewerten von Routing-Matrizen beschrieben, das heißt im Wesentlichen von Routing-Wegen. Das Bewertungskriterium ist dabei die Routing-Dauer gewesen. Da eine wirkliche kombinatorische Optimierung aufgrund der Komplexität üblicherweise nicht in kurzer Zeit durchführbar ist, wurde oben eine Alternative vorgestellt.

[0370] Der frei wählbare Parameter ist das Gewicht g. Zur (Teil)-Optimierung der Routing-Dauer kann dieser Prozess vom Portalprozessor auch mehrfach mit unterschiedlichem Gewicht g durchgeführt werden.

[0371] Üblicherweise wird man die Gewichte g = 0, 1, 2, 3, ... betrachten und untersuchen.

[0372] Diese haben sich bei numerischen Betrachtungen als vorteilhaft erwiesen. Dasjenige Routing, welches die kürzeste Routing-Dauer besitzt, kann anschließend endgültig verwendet werden.

[0373] Um den Prozess mehrfach durchführen zu können, verwendet der Portalprozessor die Nachricht MessRetry, die alle Kanäle, Farbregionen und Farbabstände löscht, wie in **Fig.** 48 dargestellt ist. Zur Terminierung des Prozesses wird die MessRetry-Nachricht mit dem Parameter "stamp" versehen, dessen Wert nicht identisch ist mit dem entsprechenden gespeicherten Parameter der Prozessoreinheit. Anders ausgedrückt verwendet der Portalprozessor bei jedem erneuten Zurücksetzen einen anderen Parameter "stamp".

[0374] Bei Empfang einer eingehenden MessRetry-Nachricht **4801** mit dem Parameter "stamp" werden von dem jeweiligen die MessRetry-Nachricht **4801** empfangenen Prozessoreinheit die folgenden Schritte durchgeführt:

1. Falls der eigene Stempelparameter identisch zu dem in der MessRetry-Nachricht enthaltenem Stempelparameter "stamp" ist, wird die Verarbeitung beendet.
2. Der eigene Stempelparameter wird auf den Wert des in der MessRetry-Nachricht enthaltenen Stempelparameterwerts "stamp" gesetzt.
3. Alle Nummerierungen, Kanäle, Farbregionen, Farbabstände und Token-Blockierungen werden gelöscht.
4. Über alle Verbindungen mit Ausnahme der Verbindung zu der die MessRetry-Nachricht sendenden Prozessoreinheit werden zusätzliche MessRetry-Nachrichten **4802** übertragen, wie in **Fig.** 48 dargestellt ist.

[0375] Während des Betriebs der Pixel-Anordnung können durch Abnutzung Fehler auftreten, die zum Zeitpunkt der oben beschriebenen Selbstorganisation noch nicht vorhanden waren. Zur Selbsterkennung dieser Fehler können weitere Nachrichten verwendet werden.

[0376] Nach den oben dargestellten Modellannahmen kann aus Sicht eines lokalen Prozessors ein Fehler nur darin bestehen, dass ein bislang verbundener Nachbar-Prozessor nicht mehr erreichbar ist. Er kann hingegen nicht beurteilen, ob nur die Verbindung zu diesem Nachbar-Prozessor oder ob der Nachbarprozessor selber ausgefallen ist. Bei einem solchen Vorkommnis kann aber eine Fehlernachricht, im Weiteren als MessError-Nachricht bezeichnet, an den Portalprozessor senden, die ihn selbst identifiziert, vorzugsweise unter Verwendung der eigenen Pixelnummer als Nachrichtenparameter und die zusätzlich die Nummer der neu ausgefallenen Verbindung enthält.

[0377] Eine mögliche Reaktion des Portalprozessors auf eine solche Nachricht ist ein globaler Reset der Pixelanordnung mit Hilfe einer MessReset-Nachricht.

[0378] Als Reaktion auf diese Nachricht leitet jeder Pixelprozessor diese Nachricht an alle Nachbar-Prozessoren weiter und löscht alle Daten, die bei der Organisation ermittelt wurden. Zur Terminierung dieses Prozesses sollte jeder Pixelprozessor eine gewisse Totzeit einhalten, vor deren Ende er nicht auf weitere Nachrichten reagiert. Die Totzeit verhindert, dass die Verbreitung der MessReset-Nachricht unendlich oft wiederholt wird.

[0379] Zusammenfassend ist in **Fig.** 51 eine Übersicht über die verwendeten Nachrichten, deren jeweiligen Parameter aufgeführt.

[0380] Es ist in diesem Zusammenhang anzumerken, dass der Nachrichtenkatalog selbstverständlich funktional um beliebige zusätzliche Nachrichten erweiterbar ist.

[0381] **Fig.** 49 zeigt ein zusätzliches Ausführungsbeispiel der Erfindung, bei dem die Prozessoreinheiten **4901** matrixförmig in einer ersten Hierarchieebene angeordnet sind und voll miteinander vermascht sind. Die Prozessoren **4901** der Pixel-Anordnung **4900** weisen in diesem Fall eine viereckige, vorzugsweise eine quadratische Form auf und sind steuernd und lesend mit jeweils einer Gruppe von Pixeln, das heißt Bildpunkten oder Sensorelementen, gekoppelt.

[0382] Gemäß diesem Ausführungsbeispiel ist jeder Prozessor zur Steuerung beziehungsweise zum Auslesen von 4 · 4 Pixeln **4902**, welche jeweils zu einem Pixelblock **4903** gruppiert sind, gekoppelt.

[0383] Die einzelnen Pixelgruppenprozessoren sind wie oben dargestellt in einem orthogonalen Netz miteinander durch lokale Nächste-Nachbar-Verbindungen **4904** miteinander vernetzt.

[0384] Zwischen jeweils zwei Prozessoren **4901** können eine oder mehrere Verbindungen verlaufen, um beispielsweise eine bidirektionale Datenübertragung oder aber die separate Verteilung der Versorgungsspannungen zu ermöglichen. In der Schicht der Pixel **4902** sind wie oben erläutert, die einzelnen Pixelelemente enthalten, wobei jeder Pixelblock **4903** mittels einer konventionellen Matrixansteuerung angesteuert werden, wie sie beispielsweise in **Fig.** 50 dargestellt ist.

[0385] Die Matrixsteuerung **5000** in **Fig.** 50 zeigt beispielhaft einen Pixelblock **5001** sowie eine Spaltenadressierungseinheit, welche als Schieberegister **5002** ausgestaltet ist sowie eine Zeilen-Adressierungseinheit **5003**, welche ebenfalls als Schieberegister ausgestaltet ist. Die Daten werden über eine Datenquelle **5004** dem Eingang des Schieberegisters **5001** der Spalten-Adressierungseinheit zugeführt sowie einer Takterzeugungseinheit **5005**, welche ausgangsseitig mit dem Eingang der Spalten-Adressierungseinheit **5003** gekoppelt ist.

[0386] Auf diese Weise wird der vertikale Verdrahtungsaufwand zwischen den beiden Ebenen, der Prozessorebene und der Pixelebene gering gehalten, da nur eine serielle Datenübertragung über nur eine Leitung durchgeführt wird, wobei auch die Taktsignale aus dem Datensignal selbst gewonnen werden.

[0387] Es ist in diesem Zusammenhang anzumerken, dass mehr als zwei Ebenen vorhanden sein können, das heißt Gruppen von Pixelgruppenprozessoren können wiederum zu Gruppen zusammengefasst werden, so dass sich eine tiefere hierarchische Baumstruktur ergibt.

[0388] Die Pixelgruppenprozessoren können wie in der Abbildung orthogonal miteinander vermascht sein, es sind aber auch andere Vermaschungen, insbesondere eine hexagonale Vermaschung wie oben erläutert ebenfalls möglich.

[0389] Die Pixelgruppenprozessoren können ferner Ihre Nachrichten in beliebigen Formaten austauschen. Vorzugsweise tauschen sie jedoch Nachrichtenblöcke miteinander aus, die zum Beispiel Adresscodes enthalten können, mit Hilfe derer die in Datenpakete gefassten Bildinformationen anschließend von Prozessor zu Prozessor geführt werden können. Auf diese Weise können auch Defekte in der Matrix umgangen werden, was bei einer starren x/y-Adressierung über Zeilenleitungen und Spaltenleitungen nicht möglich ist.

[0390] Je nach Leistungsfähigkeit der Pixelgruppenprozessoren können diese neben der Informationsübertragung und Informationsverteilung auch noch andere Aufgaben übernehmen, wie beispielsweise eine Kompression oder Dekompression von Bildinformation.

[0391] Die Architektur kann auch für großflächige Sensorarrays, wie beispielsweise Fingerprint-Sensoren, Touchpads oder Schrifterkennungs-Sensoren verwendet werden.

[0392] So kann zum Beispiel jeder Pixelgruppenprozessor darauf warten, bis ein Ereignis, das heißt eine Eingabe mit einem Stift etc. in einer Pixelgruppe auftritt und diese Daten dann zum Rand der jeweiligen Matrix senden beziehungsweise routen. Auf diese Weise kann vermieden werden, dass stets die gesamte Sensorfläche von einer externen Adressierungseinheit abgescannt werden muss. Anschaulich entspricht dies einer lokalen Triggerung durch ein Ereignis an Stelle einer globalen Bildauswertung.

[0393] Die Pixelgruppenprozessoren können beispielsweise mit Hilfe einer Technik wie beispielsweise der Fluidic Self Assembly, wie sie in den [3] beschrieben ist, in ein geeignetes Trägersubstrat eingebracht werden, wie gemäß dem ersten Ausführungsbeispiel beschrieben.

[0394] Die Verbindungen zwischen den Prozessoren sowie die Sensor oder Displaymatrizen können dann in weiteren Prozessschritten auf das Trägersubstrat mit den Prozessoren aufgebracht werden.

[0395] Vorzugsweise können die oberen Funktionsschichten durch druckbare Schaltungen realisiert werden, wie sie bei der Polymerelektronik vorgesehen sind.

[0396] Es ist anzumerken, dass die oben beschriebenen Teil-Verfahren alle unabhängig voneinander, d. h. selbständig durchgeführt werden können. Sie benötigen jeweils nur die für das jeweilige Teil-Verfahren erforderlichen Eingangsinformationen. Es ist jedoch grundsätzlich unerheblich, auf welche Weise die erforderlichen Eingangsinformationen ermittelt worden sind. Dennoch ist auf die erheblichen Vorteile im Rahmen der Verzahnung und gemeinsamen Durchführung mehrerer oder aller Teil-Verfahren hinzuweisen, da in diesem Fall das erfindungsgemäße oben beschriebene Konzept durchgängig implementiert ist.

[0397] In diesem Dokument sind folgende Veröffentlichungen zitiert:

[1] T. J. Nelson und J. R. Wullert, Electronic Information Display Technologies, World Scientific, Kapitel 8.2, 1997

[2] K. Amundson, Microencapsulated Electrophoretic Materials for Electronic Paper Displays, International Display Research Conference, 2000

[3] J. S. Smith, High-Density, low parasitic direct Integration by Fluidic Self-Assembly (FSA), IEDM, S. 201–204, 2000

Bezugszeichenliste

100 Pixel-Anordnung
101 Substrat
102 Vertiefung
103 Prozessoreinheit
300 Bildanzeigendes System
301 Quelle
302 Anzeigeeinheits-Portal
303 Pixel-Anordnung
401 Bidirektionale Kommunikationsschnittstellen
402 Elektrische Leitung
501 Bidirektionale Kommunikationsschnittstellen
502 Elektrische Leitung
600 Erste Ausrichtung
601 Zweite Ausrichtung
603 Dritte Ausrichtung
604 Vierte Ausrichtung
605 Fünfte Ausrichtung
606 Sechste Ausrichtung
700 Gerichteter Graph
701 Ungerichteter Graph
800 Gerichteter Baum
900 Ungerichteter Graph
901 Gerichteter Pixel-Anordnungs-Graph
902 Portal-Knoten
903 Knoten
904 Zuleitung
905 Kante
1000 Zulässiger Baum
1001 Portal-Knoten
1100 Baum
1201 Nachricht
1202 Portal-Knoten
1203 Einleit-Pixelprozessoren
1204 Erste innere Knoten
1401 Erster Pixelprozessor
1402 Zweiter Pixelprozessor
1403 Bidirektionale Kommunikationsschnittstelle erster Pixelprozessor
1404 Bidirektionale Kommunikationsschnittstelle zweiter Pixelprozessor
1405 Zuleitung
1406 Erste Nachricht
1407 Zweite Nachricht
1500 Prozessoreinheit
1501 MessKoherenz-Nachricht
1601 MessKoherenz-Nachricht
1602 MessKoherenz-Nachricht
1603 MessKoherenz-Nachricht
1604 MessKoherenz-Nachricht

1605 MessKoherenz-Nachricht
1606 MessKoherenz-Nachricht
1701 MessPosition-Nachricht
1702 MessPosition-Nachricht
1703 MessPosition-Nachricht
1704 MessPosition-Nachricht
1705 MessPosition-Nachricht
1706 MessPosition-Nachricht
1800 Prozessor-Anordnung
1801 Pixelprozessor
1901 MessDistance-Nachricht
1902 MessDistance-Nachricht
1903 MessDistance-Nachricht
1904 MessDistance-Nachricht
1905 MessDistance-Nachricht
1906 MessDistance-Nachricht
2001 Prozessoreinheit
2002 Unterste Zeile Prozessor-Anordnung
2003 Südwest-Seite Prozessoreinheit
2100 Prozessor-Anordnung
2101 Unterste Zeile Prozessor-Anordnung
2102 Prozessoreinheiten, welche nicht mit dem Portalprozessor gekoppelt sind
2103 Prozessoreinheiten, welche mit dem Portalprozessor gekoppelt sind
2201 MessOrganize-Nachricht
2202 MessOrganize-Nachricht
2203 MessOrganize-Nachricht
2204 MessOrganize-Nachricht
2205 MessOrganize-Nachricht
2206 MessOrganize-Nachricht
2600 Horizontalriss
2700 Horizontalriss
2801 Eingehende MessCountNodes-Nachricht
2802 Gesendete MessCountNodes-Nachricht
2901 Erste eingehende MessNodesSize-Nachricht
2902 Zweite eingehende MessNodesSize-Nachricht
2903 Gesendete MessNodesSize-Nachricht
3201 MessColDistance-Nachricht
3202 MessColDistance-Nachricht
3203 MessColDistance-Nachricht
3204 MessColDistance-Nachricht
3205 MessColDistance-Nachricht
3206 MessColDistance-Nachricht
3301 Empfangene MessBlockToken-Nachricht
3302 Gesendete MessBlockToken-Nachricht
3401 Eingehende MessToken-Nachricht
3402 Gesendete MessBlockToken-Nachricht
3403 Gesendete MessBlockToken-Nachricht
3404 Gesendete MessBlockToken-Nachricht
3405 Gesendete MessBlockToken-Nachricht
3406 Gesendete MessBlockToken-Nachricht
3601 Eingehende MessDeletechannels-Nachricht
3602 Gesendete MessDeletechannels-Nachricht
3603 Gesendete MessDeletechannels-Nachricht
3604 Gesendete MessDeletechannels-Nachricht
3605 Gesendete MessDeletechannels-Nachricht
3606 Gesendete MessDeletechannels-Nachricht
3701 Eingehende MessColOrganize-Nachricht
3702 Gesendete MessColOrganize-Nachricht
3703 Gesendete MessColOrganize-Nachricht
3704 Gesendete MessColOrganize-Nachricht
3705 Gesendete MessColOrganize-Nachricht
3706 Gesendete MessColOrganize-Nachricht
3801 Mäander-Weg
3901 Mäander-Weg
4301 Eingehende MessNumbering-Nachricht
4302 Gesendete MessNumbering-Nachricht
4600 Routing-Tabelle
4801 Eingehende MessRetry-Nachricht

4802 Gesendete MessRetry-Nachricht
4900 Pixel-Anordnung
4901 Prozessoreinheit
4902 Pixel
4903 Pixelblock
5000 Matrixsteuerung
5001 Pixelblock
5002 Schieberegister
5003 Zeilen-Adressierungseinheit
5004 Datenquelle
5005 Takterzeugungseinheit

Patentansprüche

1. Verfahren zum Bestimmen eines Abstands von Prozessoreinheiten zu mindestens einer Referenzposition in einer Prozessor-Anordnung mit einer Vielzahl von Prozessoreinheiten, wobei jede Prozessoreinheit über eine bidirektionale Kommunikationsschnittstelle mit mindestens einer benachbarten Prozessoreinheit gekoppelt ist und wobei Nachrichten ausgetauscht werden zwischen einander benachbarten Prozessoreinheiten,
bei dem eine erste Nachricht von einer ersten Prozessoreinheit erzeugt wird, wobei die erste Nachricht eine erste Abstandsinformation enthält, welche den Abstand der ersten Prozessoreinheit oder den Abstand einer die erste Nachricht empfangenden zweiten Prozessoreinheit von der Referenzposition enthält,
bei dem die erste Nachricht von der ersten Prozessoreinheit zu der zweiten Prozessoreinheit gesendet wird,
bei dem abhängig von der Abstandsinformation der Abstand der zweiten Prozessoreinheit von der Referenzposition ermittelt oder gespeichert wird, und
bei dem von der zweiten Prozessoreinheit eine zweite Nachricht erzeugt wird, welche eine zweite Abstandsinformation enthält, welche den Abstand der zweiten Prozessoreinheit oder den Abstand einer die zweite Nachricht empfangenden dritten Prozessoreinheit von der Referenzposition enthält,
bei dem die zweite Nachricht von der zweiten Prozessoreinheit zu der dritten Prozessoreinheit gesendet wird,
bei dem abhängig von der zweiten Abstandsinformation der Abstand der dritten Prozessoreinheit von der Referenzposition ermittelt oder gespeichert wird,
bei dem die Verfahrensschritte für alle Prozessoreinheiten in der Pixel-Anordnung durchgeführt werden.

2. Verfahren nach Anspruch 1, bei dem jede Prozessoreinheit mit mindestens einem Pixel gekoppelt ist, so dass das Pixel von der jeweiligen ihm zugeordneten Prozessoreinheit steuerbar ist.

3. Verfahren nach Anspruch 2, bei dem zumindest ein Teil der Pixel jeweils als ein Sensor ausgestaltet ist.

4. Verfahren nach Anspruch 2, bei dem zumindest ein Teil der Pixel jeweils als ein bildgebendes Element ausgestaltet ist.

5. Verfahren nach einem der Ansprüche 1 bis 4,
bei dem die Prozessoreinheiten jeweils in einem hexagonalen Bereich angeordnet sind, und
bei dem jede Prozessoreinheit jeweils sechs benachbarte Prozessoreinheiten aufweist, welche jeweils über eine bidirektionale Kommunikationsschnittstelle mit der Prozessoreinheit gekoppelt sind.

6. Verfahren nach einem der Ansprüche 1 bis 5, bei dem die Pixel eine hexagonale Form aufweisen.

7. Verfahren nach einem der Ansprüche 1 bis 4,
bei dem die Prozessoreinheiten jeweils in einem rechteckigen Bereich angeordnet sind, und
bei dem jede Prozessoreinheit jeweils vier benachbarte Prozessoreinheiten aufweist, welche jeweils über eine bidirektionale Kommunikationsschnittstelle mit der Prozessoreinheit gekoppelt sind.

8. Verfahren nach einem der Ansprüche 1 bis 4 oder 7, bei dem die Pixel eine rechteckige Form aufweisen.

9. Verfahren nach einem der Ansprüche 1 bis 8, bei dem vor Bestimmen des Abstandes der Pixelprozessoren von der Referenzposition die örtlichen Positionen der Prozessoreinheiten innerhalb der Prozessor-Anordnung ermittelt werden, indem ausgehend von einer Prozessoreinheit an einer Einleitstelle der Prozessor-Anordnung jeweils Positionsermittlungs-Nachrichten, welche zumindest einen Zeilenparameter z und einen Spaltenparameter s aufweisen, welche die Zeilennummer bzw. Spaltennummer der die Nachricht sendenden Prozessoreinheit oder die Zeilennummer bzw. Spaltennummer der die Nachricht empfangenden Prozessoreinheit innerhalb der Prozessor-Anordnung enthält, an benachbarte Prozessoreinheiten übermittelt werden und von der jeweiligen Prozessoreinheit die folgenden Schritte durchgeführt werden:

- falls der Zeilenparameter in der empfangenen Nachricht größer ist als die bisher gespeicherte Zeilennummer der Prozessoreinheit, so wird der eigenen Zeilennummer der Prozessoreinheit der Zeilenparameterwert z der empfangenen Nachricht zugeordnet,
- falls der Spaltenparameter in der empfangenen Nachricht größer ist als die eigene Spaltennummer der Prozessoreinheit, so wird der gespeicherten Spaltennummer der Zeilenparameterwert der empfangenen Nachricht zugeordnet,
- falls die eigene Zeilennummer und/oder die eigene Spaltennummer aufgrund der oben dargestellten Verfahrensschritte verändert worden sind, so werden neue Positionsmess-Nachrichten mit neuen Zeilenparametern und neuen Spaltenparametern erzeugt, welche jeweils die Zeilennummer und Spaltennummer der die Nachricht sendenden Prozessoreinheit oder die Zeilennummer und Spaltennummer der die Nachricht empfangenden Prozessoreinheit enthält, und diese werden über die bidirektionalen Kommunikationsschnittstellen an eine jeweilige Nachbar-Prozessoreinheit übertragen.

10. Verfahren nach einem der Ansprüche 1 bis 9,
bei dem in einem iterativen Verfahren der eigene Abstandswert der Prozessoreinheit dann verändert wird, wenn der

bisher gespeicherte Abstandswert größer ist als der um einen vorgegebenen Wert erhöhte empfangene Abstandswert in der jeweils empfangenen Nachricht, und

bei dem für den Fall, dass eine Prozessoreinheit den eigenen Abstandswert verändert, diese eine Abstandsmess-Nachricht erzeugt und über alle Kommunikationsschnittstellen an benachbarte Prozessoreinheiten sendet, wobei die Abstandsmess-Nachricht jeweils den eigenen Abstand als Abstandsinformation enthält oder den Abstandswert, den die empfangene Prozessoreinheit von dem Portalprozessor aufweist.

11. Verfahren nach Anspruch 10, bei dem der Abstandswert einen um einen vorgegebenen Wert erhöhten Wert gegenüber dem eigenen Abstandswert.

12. Prozessor-Anordnung mit einer Vielzahl von Prozessoreinheiten,

wobei jede Prozessoreinheit über eine bidirektionale Kommunikationsschnittstelle mit mindestens einer benachbarten Prozessoreinheit gekoppelt ist und wobei zum Ermitteln des jeweiligen Abstands einer Prozessoreinheit der Prozessor-Anordnung von einer Referenzposition Nachrichten ausgetauscht werden zwischen einander benachbarten Prozessoreinheiten,

wobei jede Nachricht eine Abstandsinformation enthält, welche den Abstand einer die Nachricht sendenden Prozessoreinheit oder den Abstand einer die Nachricht empfangenden Prozessoreinheit von der Referenzposition angibt, und

wobei jede Prozessoreinheit derart eingerichtet ist, dass aus der Abstandsinformation einer empfangenen Nachricht der eigene Abstand zu der Referenzposition ermittelbar ist oder speicherbar ist.

Hierzu 28 Seite(n) Zeichnungen

- Leerseite -

Nummer: DE 101 58 784 A1
Int. Cl.[7]: G 06 F 15/173
Offenlegungstag: 7. August 2003

FIG 1

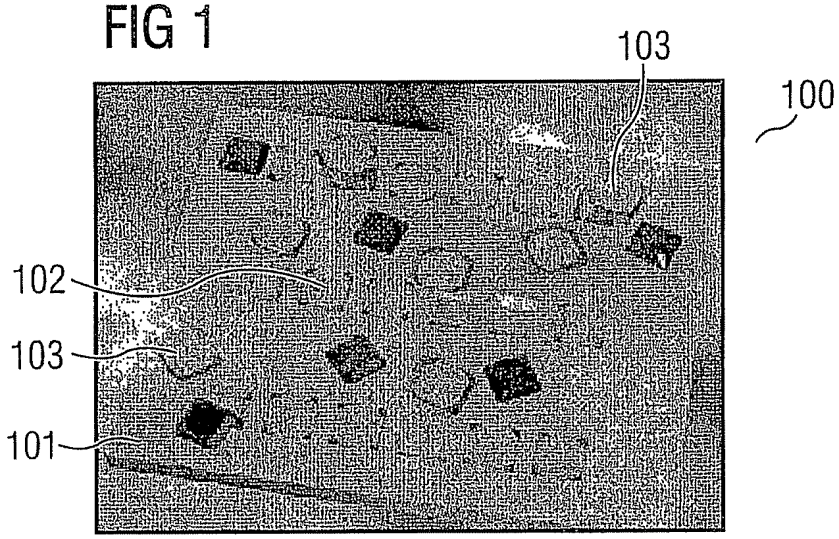


FIG 2A

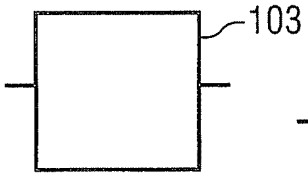


FIG 2B

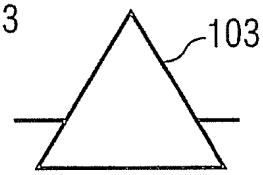


FIG 2C

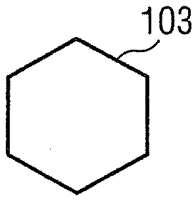


FIG 2D

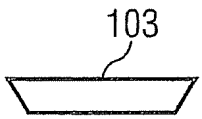
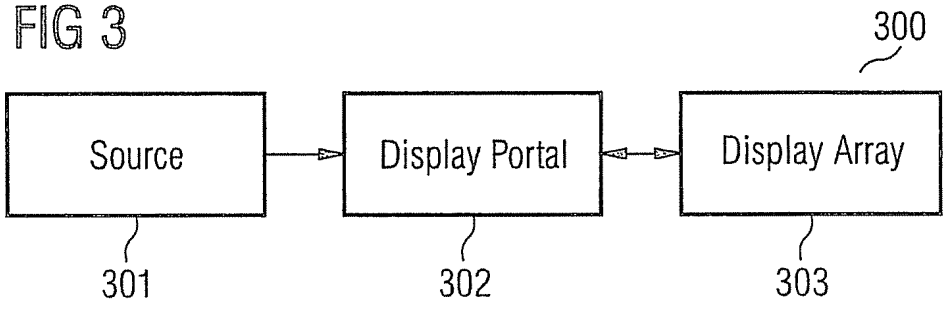


FIG 3



Nummer: DE 101 58 784 A1
Int. Cl.[7]: G 06 F 15/173
Offenlegungstag: 7. August 2003

FIG 4

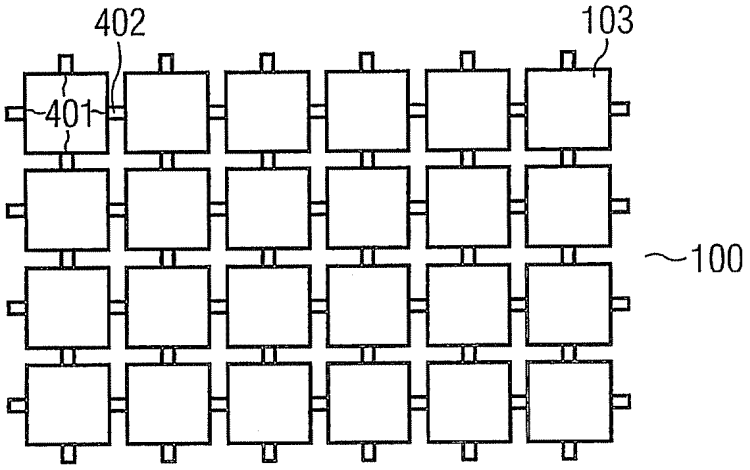


FIG 5

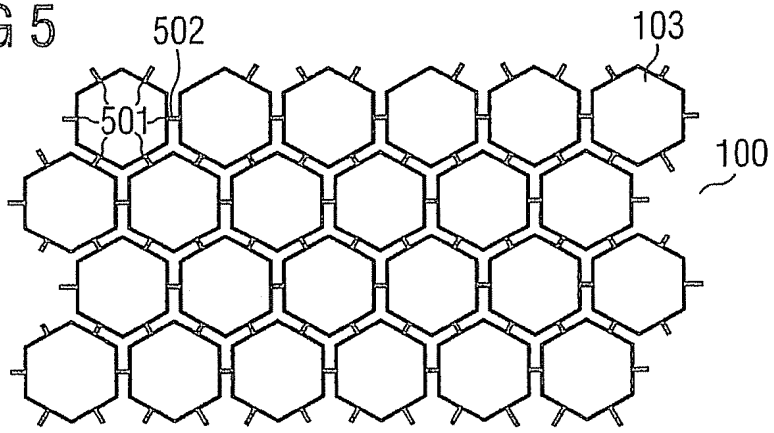
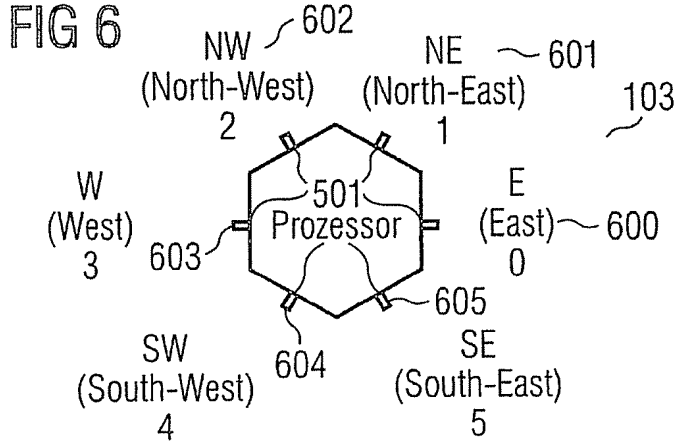


FIG 6



Nummer: DE 101 58 784 A1
Int. Cl.[7]: G 06 F 15/173
Offenlegungstag: 7. August 2003

FIG 7A

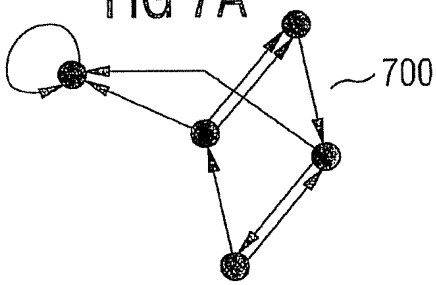


FIG 7B

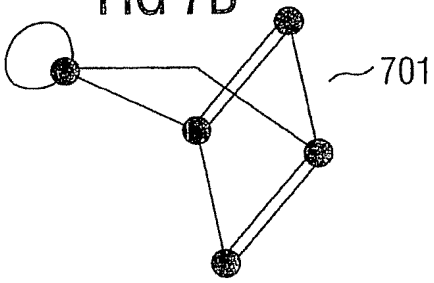


FIG 8

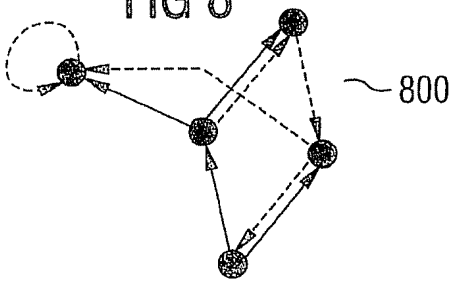


FIG 9A

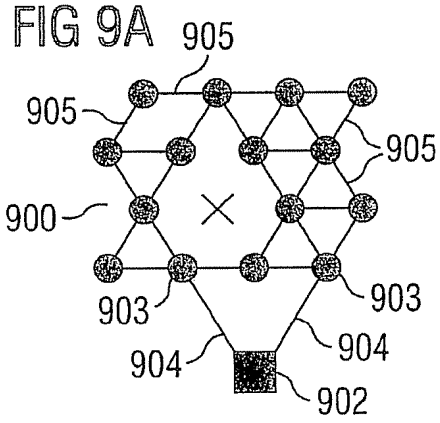


FIG 9B

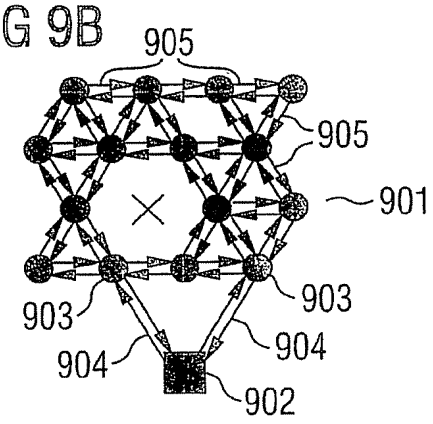


FIG 10

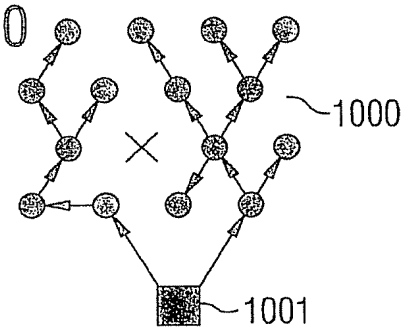
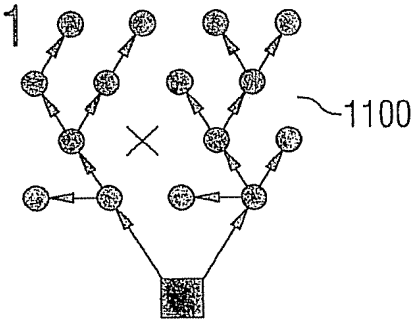


FIG 11



Nummer: DE 101 58 784 A1
Int. Cl.[7]: G 06 F 15/173
Offenlegungstag: 7. August 2003

FIG 12A

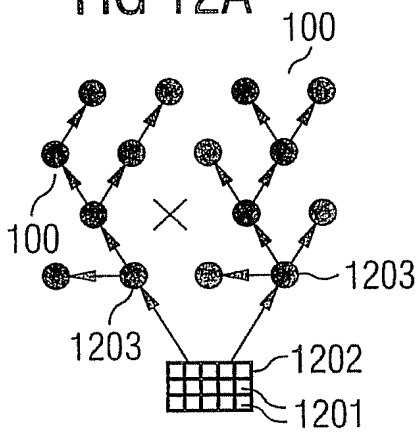


FIG 12B

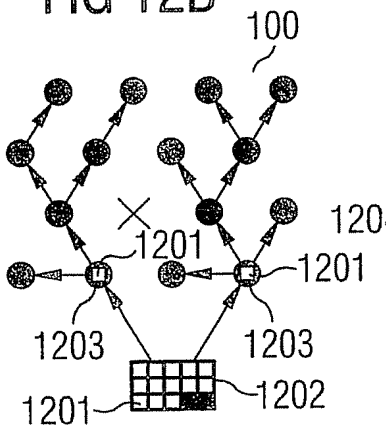


FIG 12C

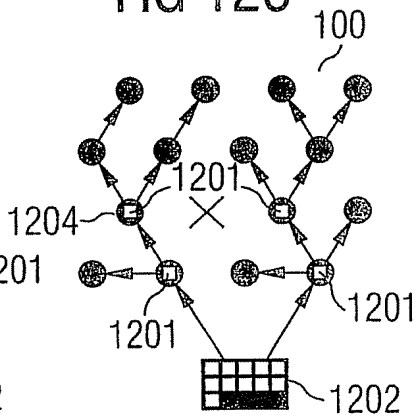


FIG 12D

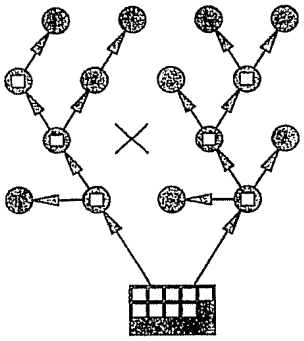


FIG 12E

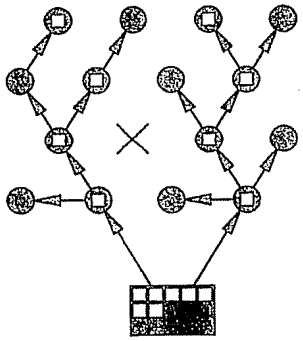


FIG 12F

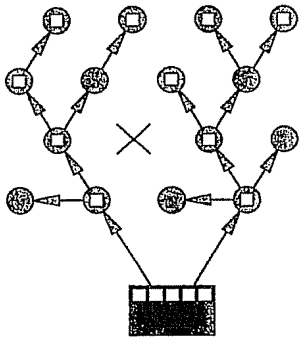


FIG 12G

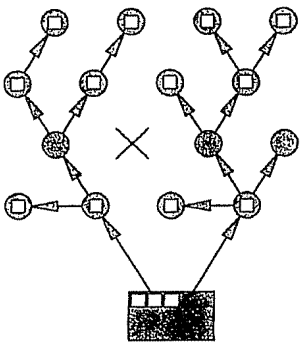


FIG 12H

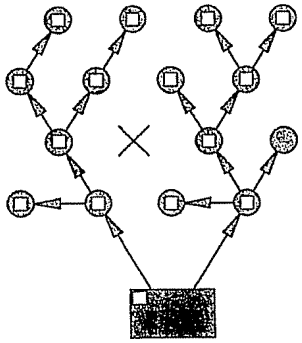
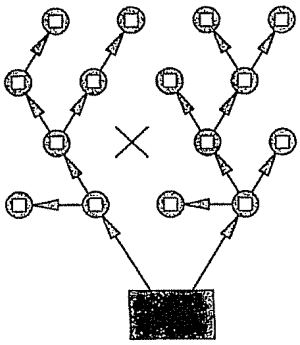


FIG 12J



Nummer: DE 101 58 784 A1
Int. Cl.[7]: G 06 F 15/173
Offenlegungstag: 7. August 2003

FIG 13A

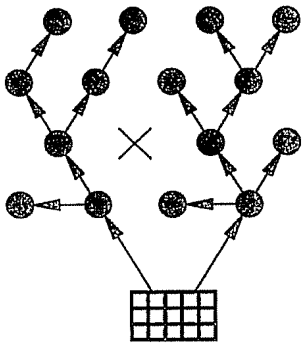


FIG 13B

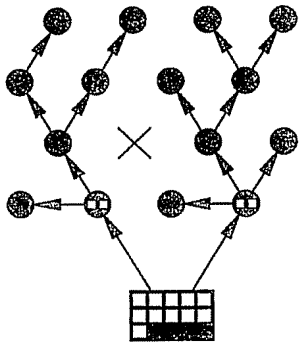


FIG 13C

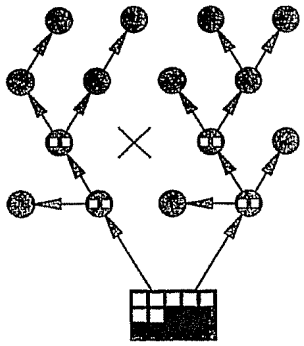


FIG 13D

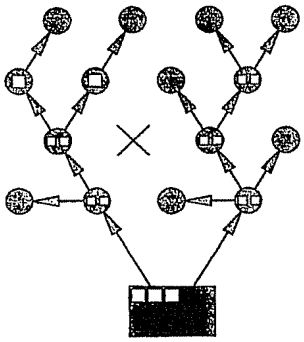


FIG 13E

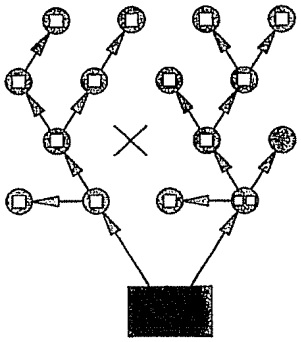


FIG 13F

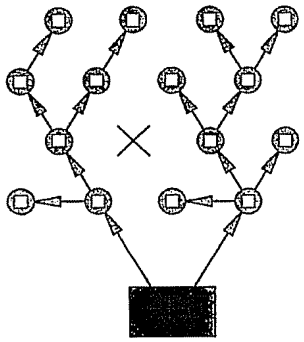
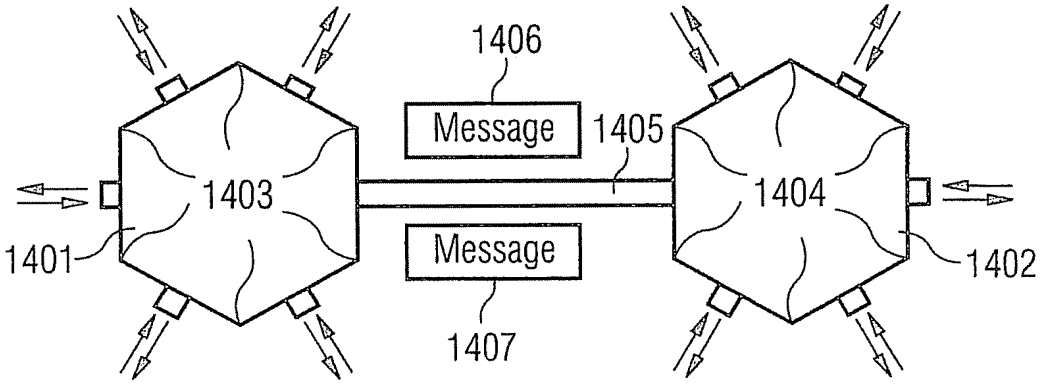


FIG 14



Nummer: DE 101 58 784 A1
Int. Cl.[7]: G 06 F 15/173
Offenlegungstag: 7. August 2003

FIG 15

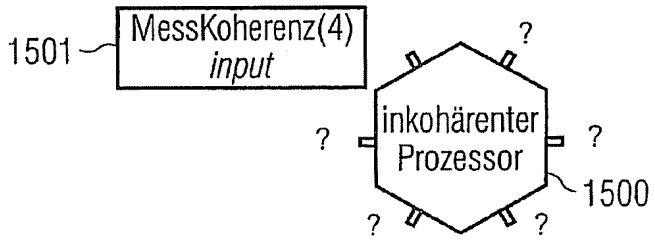


FIG 16

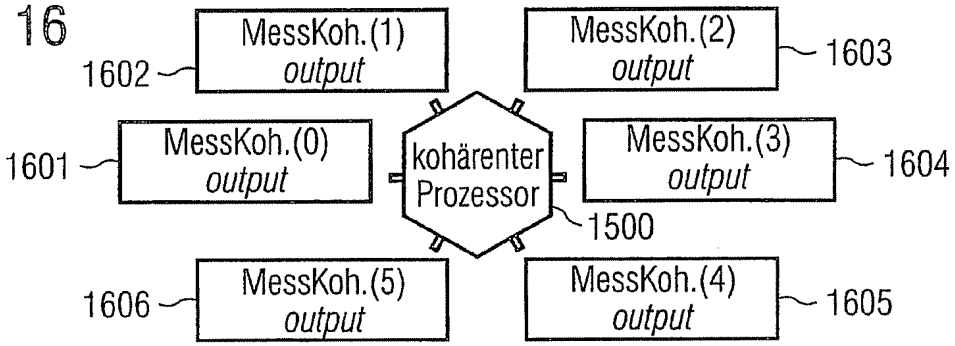


FIG 17

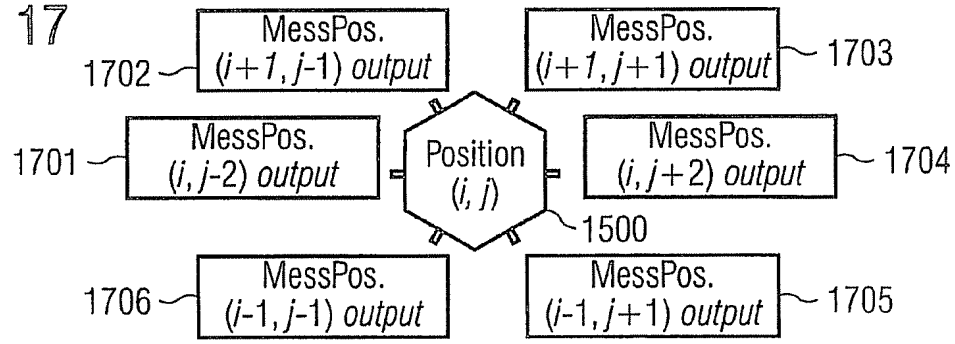
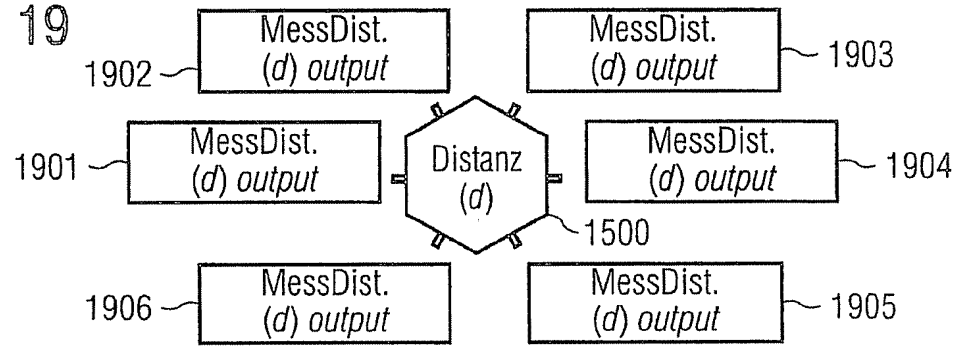


FIG 19



Nummer: DE 101 58 784 A1
Int. Cl.[7]: G 06 F 15/173
Offenlegungstag: 7. August 2003

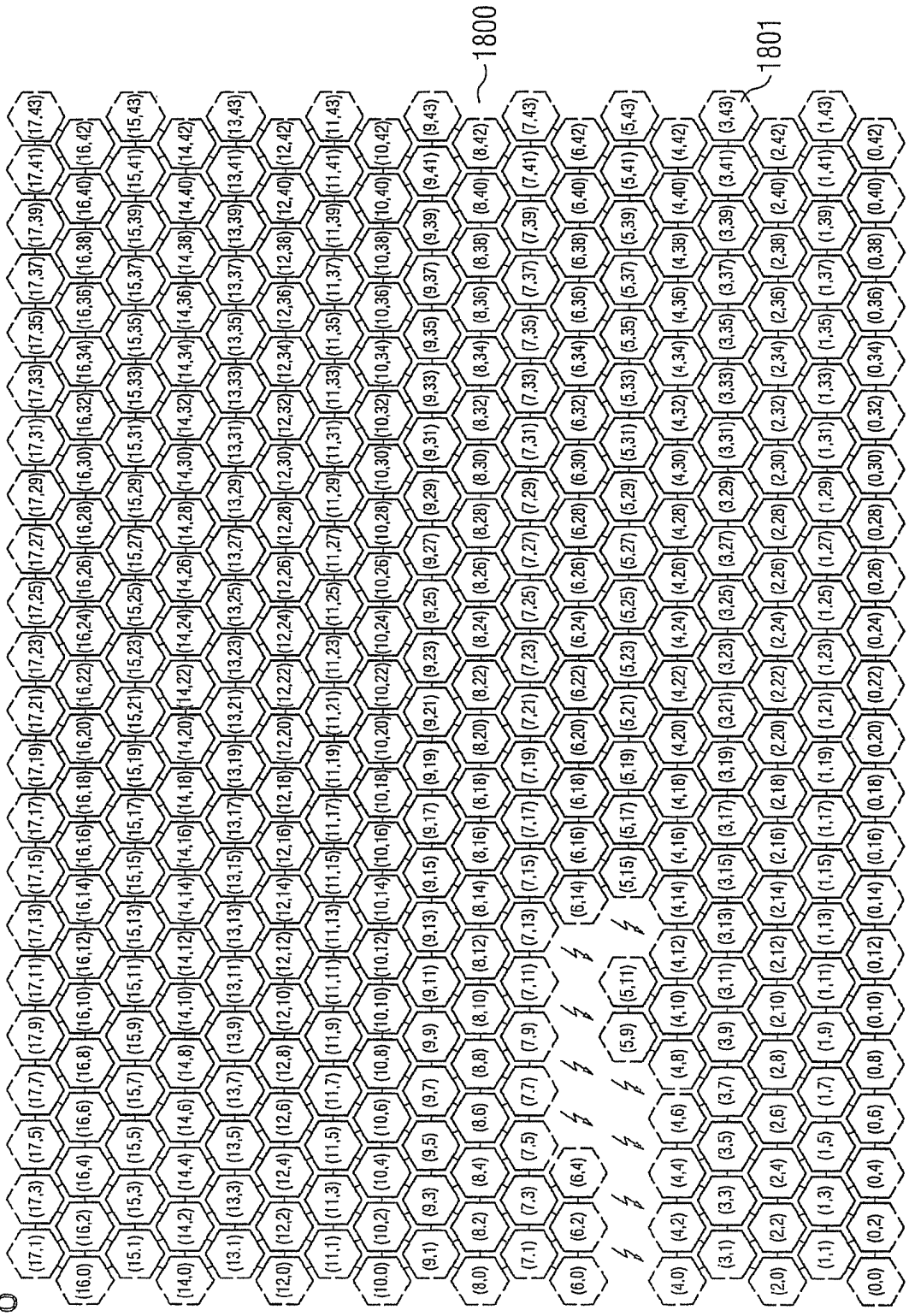


FIG 18

Nummer: DE 101 58 784 A1
Int. Cl.[7]: G 06 F 15/173
Offenlegungstag: 7. August 2003

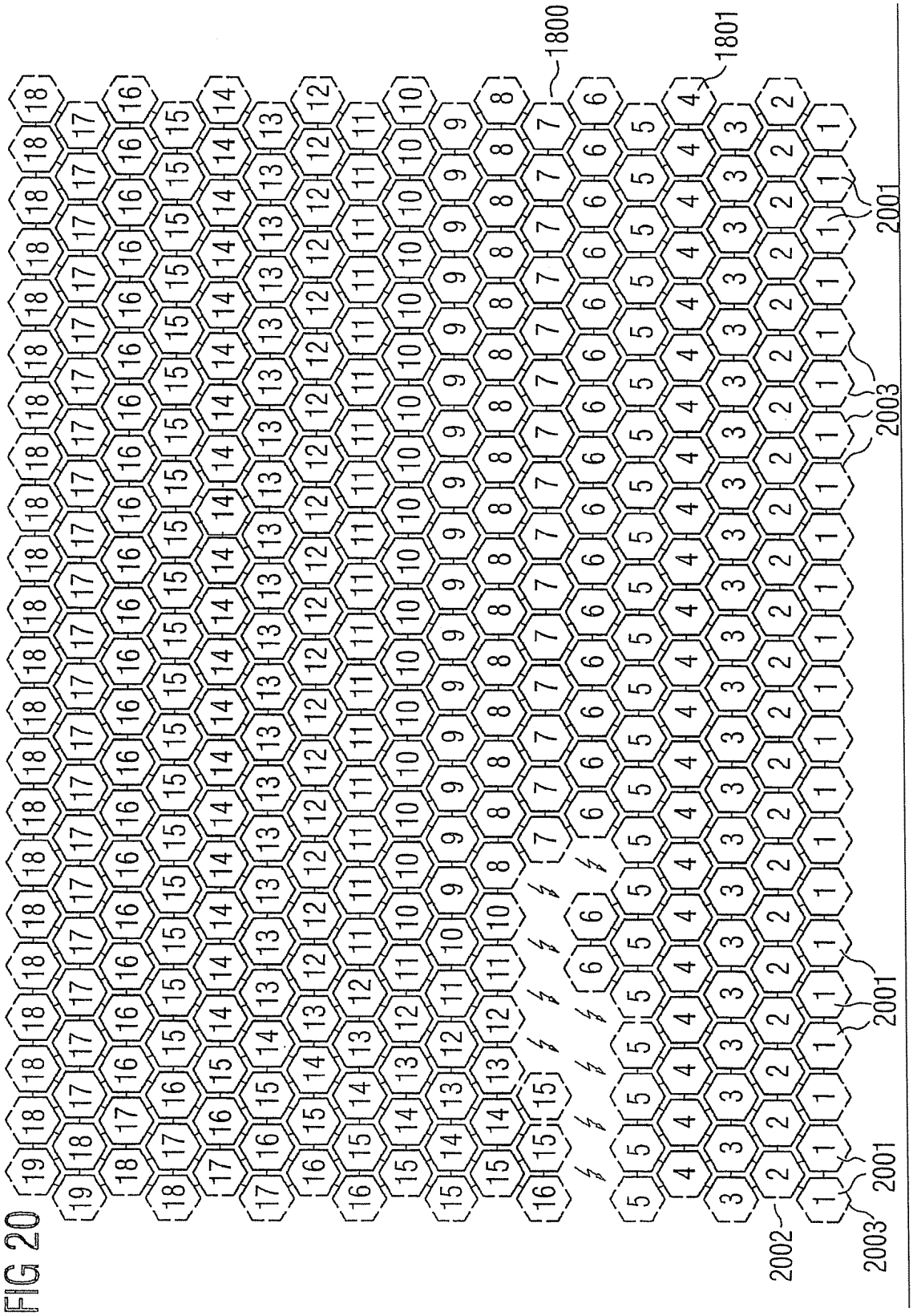


FIG 20

Nummer: DE 101 58 784 A1
Int. Cl.[7]: G 06 F 15/173
Offenlegungstag: 7. August 2003

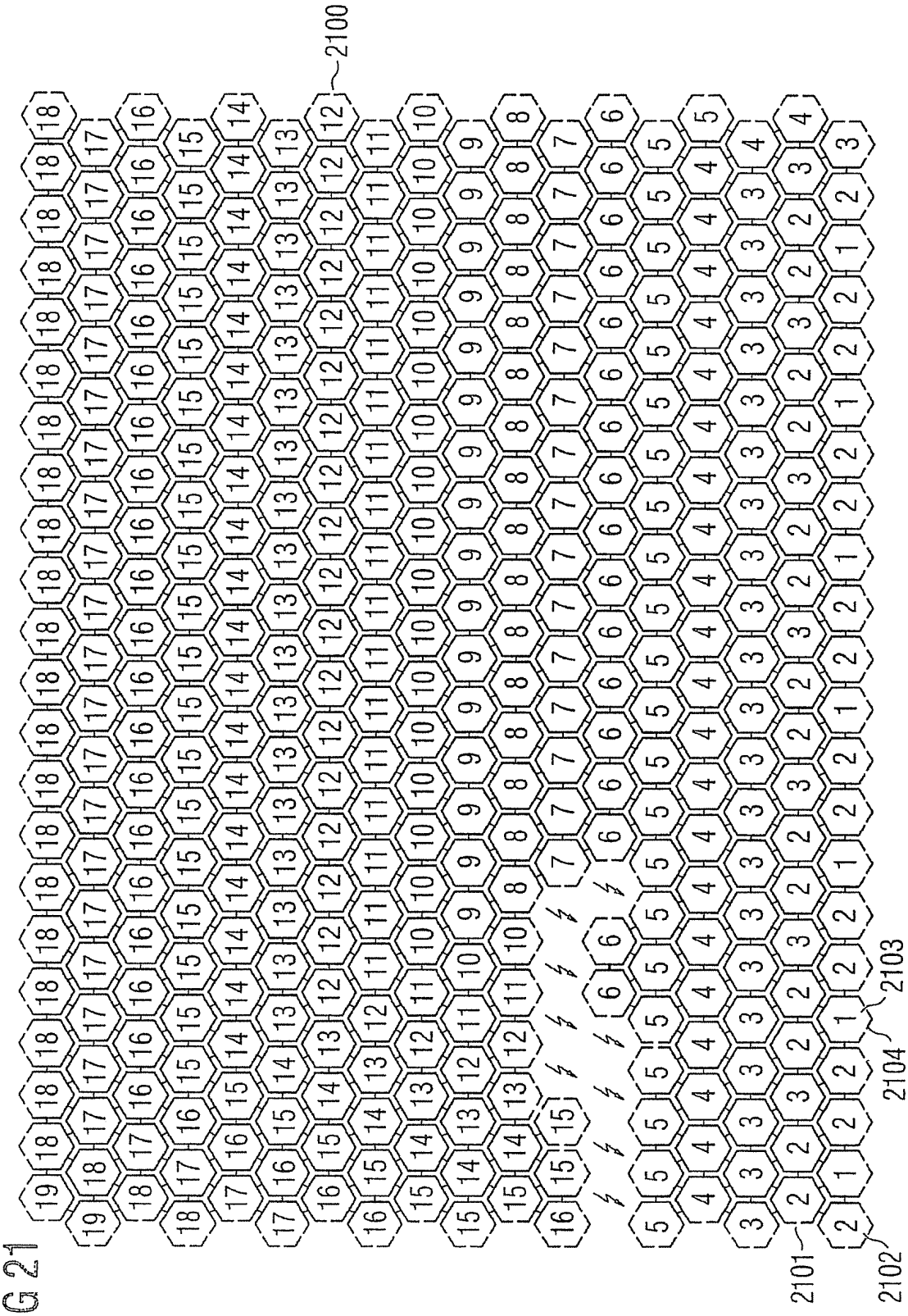


FIG 21

Nummer: DE 101 58 784 A1
Int. Cl.[7]: G 06 F 15/173
Offenlegungstag: 7. August 2003

FIG 22

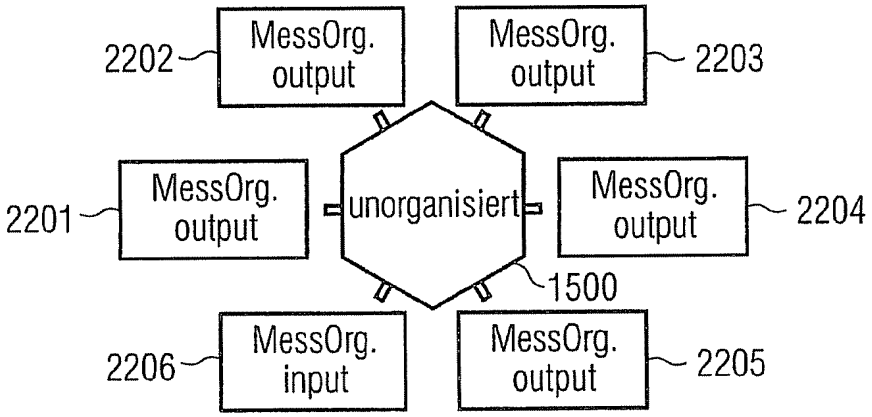


FIG 23

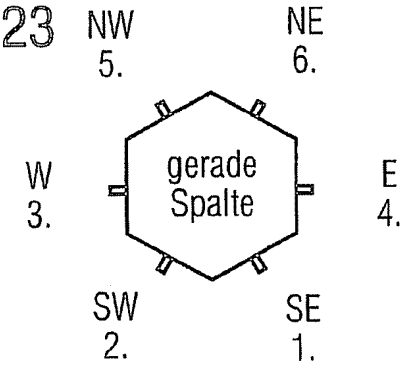


FIG 24

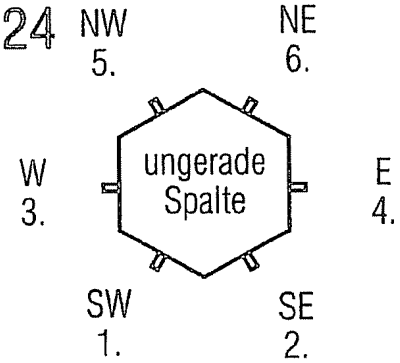
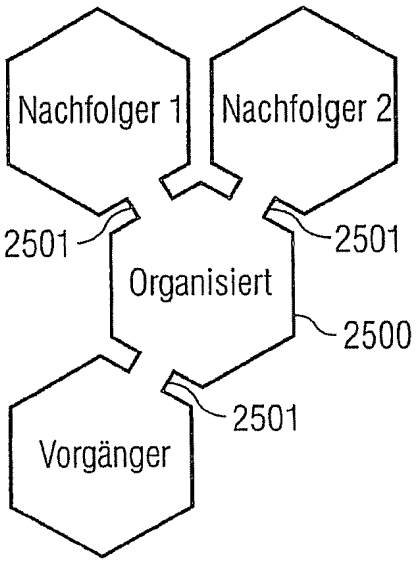


FIG 25



Nummer: DE 101 58 784 A1
Int. Cl.[7]: G 06 F 15/173
Offenlegungstag: 7. August 2003

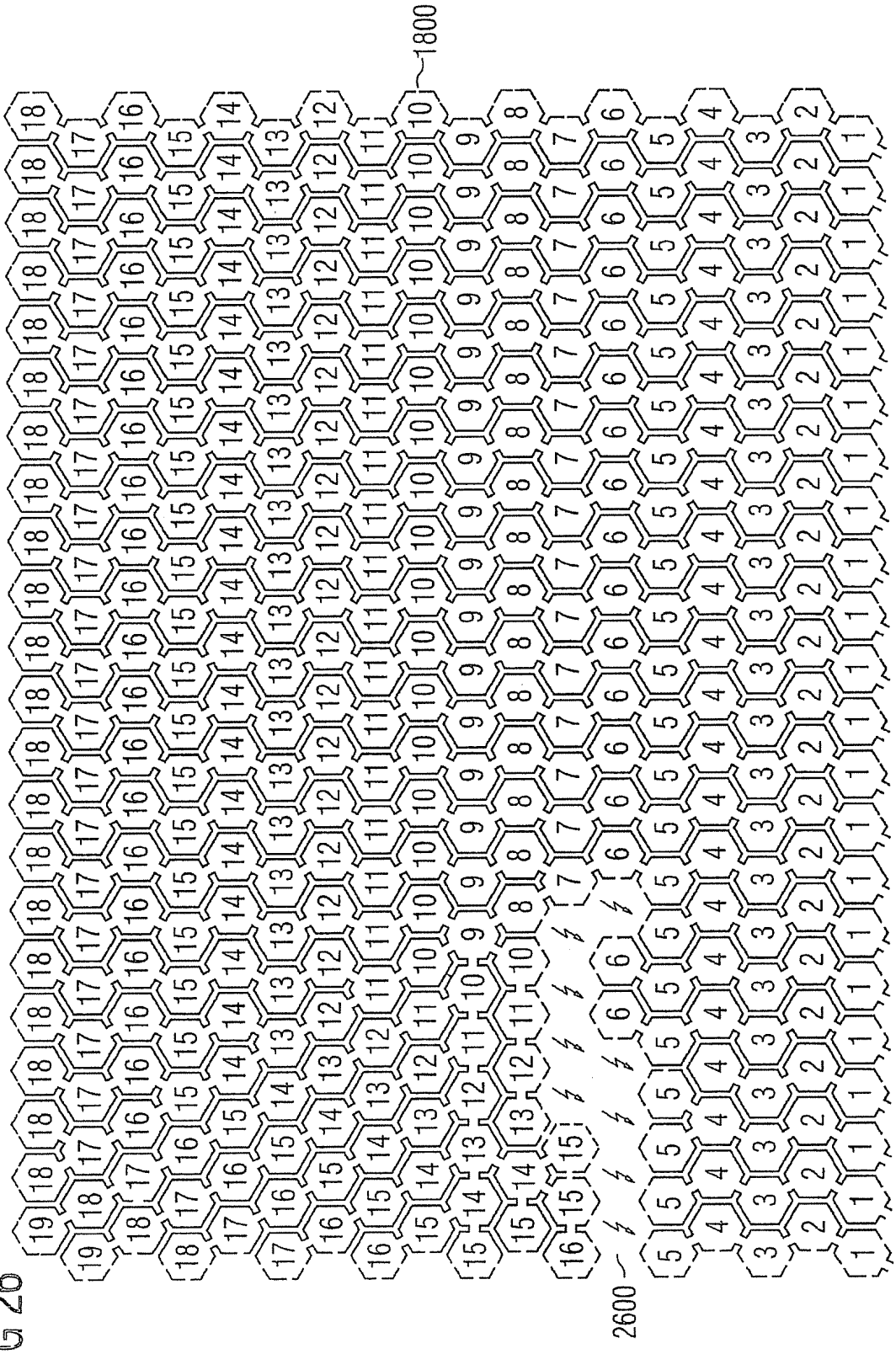


FIG 26

Nummer: DE 101 58 784 A1
Int. Cl.[7]: G 06 F 15/173
Offenlegungstag: 7. August 2003

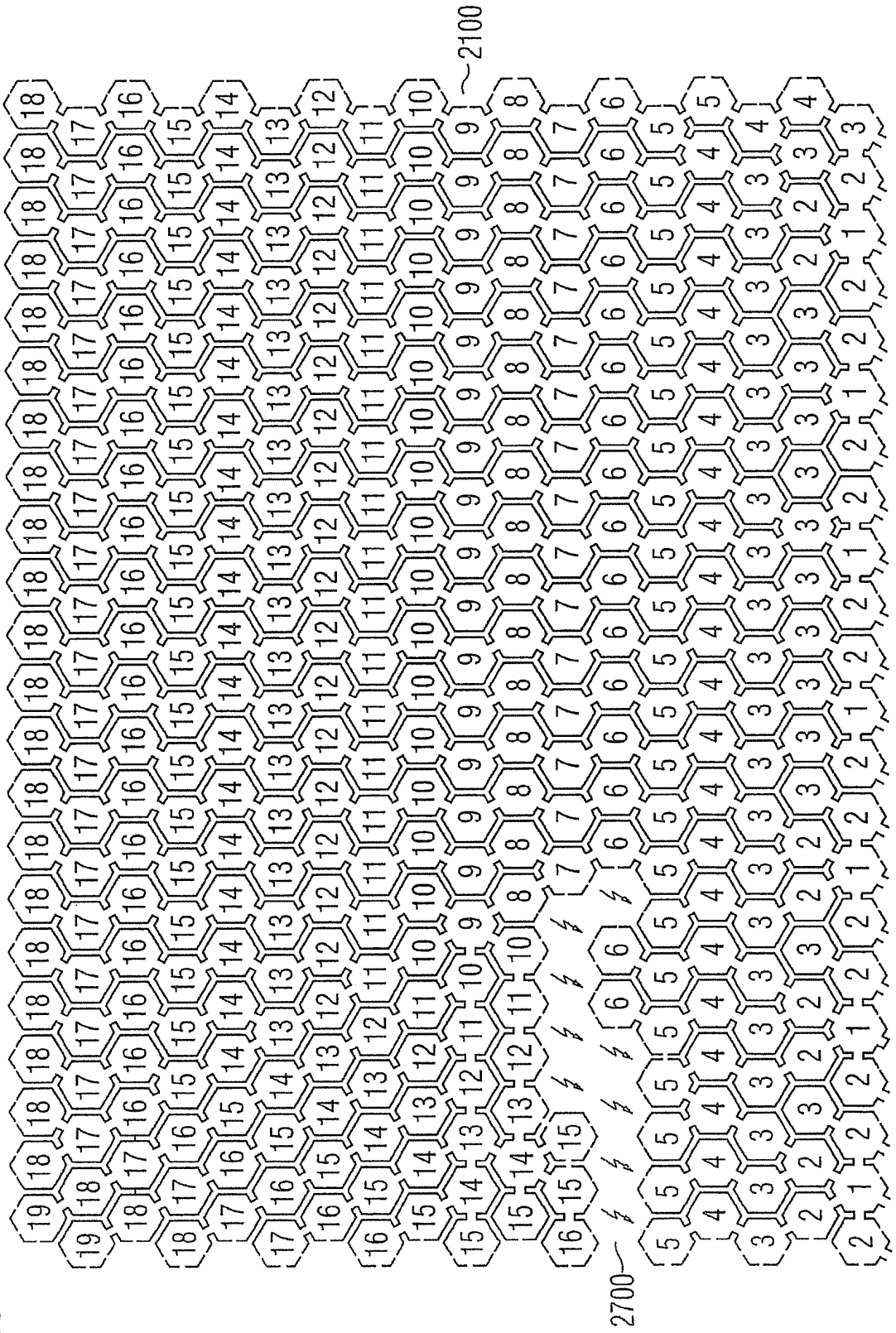


FIG 27

Nummer: DE 101 58 784 A1
Int. Cl.[7]: G 06 F 15/173
Offenlegungstag: 7. August 2003

FIG 28

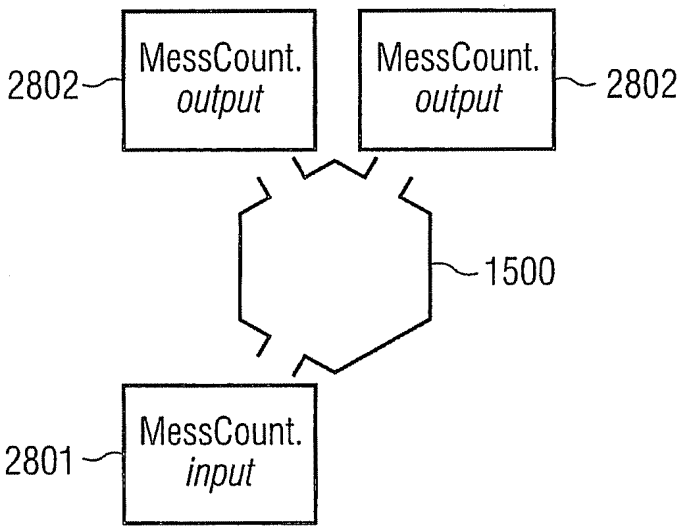
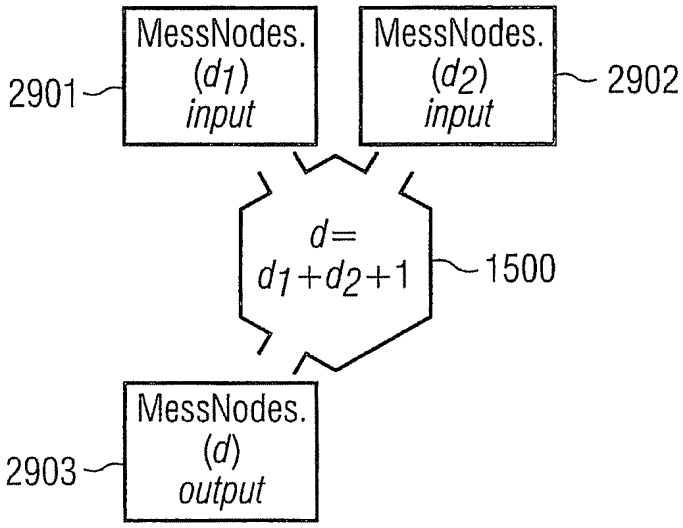


FIG 29



Nummer: DE 101 58 784 A1
Int. Cl.[7]: G 06 F 15/173
Offenlegungstag: 7. August 2003

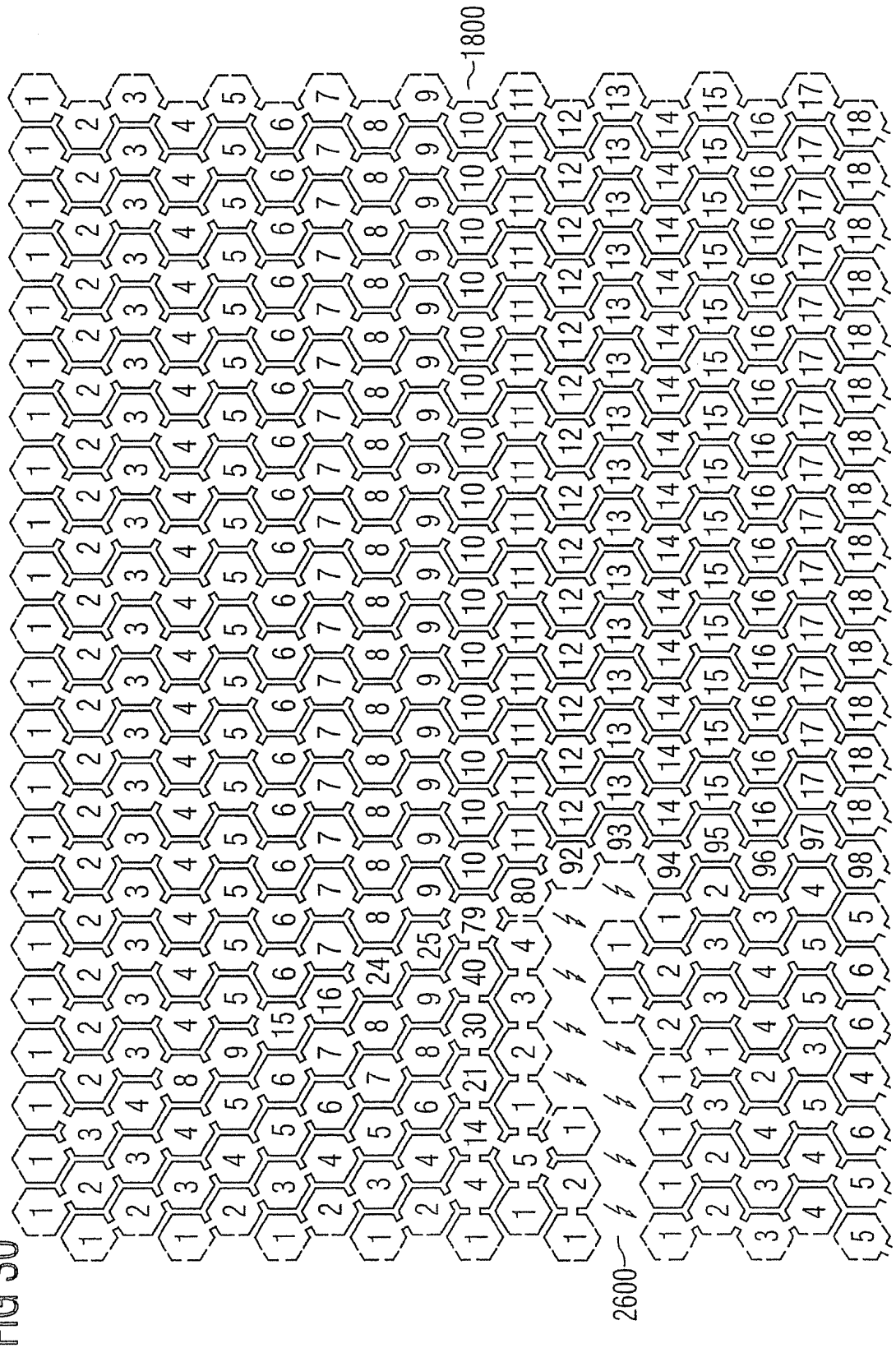


FIG 30

Nummer: DE 101 58 784 A1
Int. Cl.[7]: G 06 F 15/173
Offenlegungstag: 7. August 2003

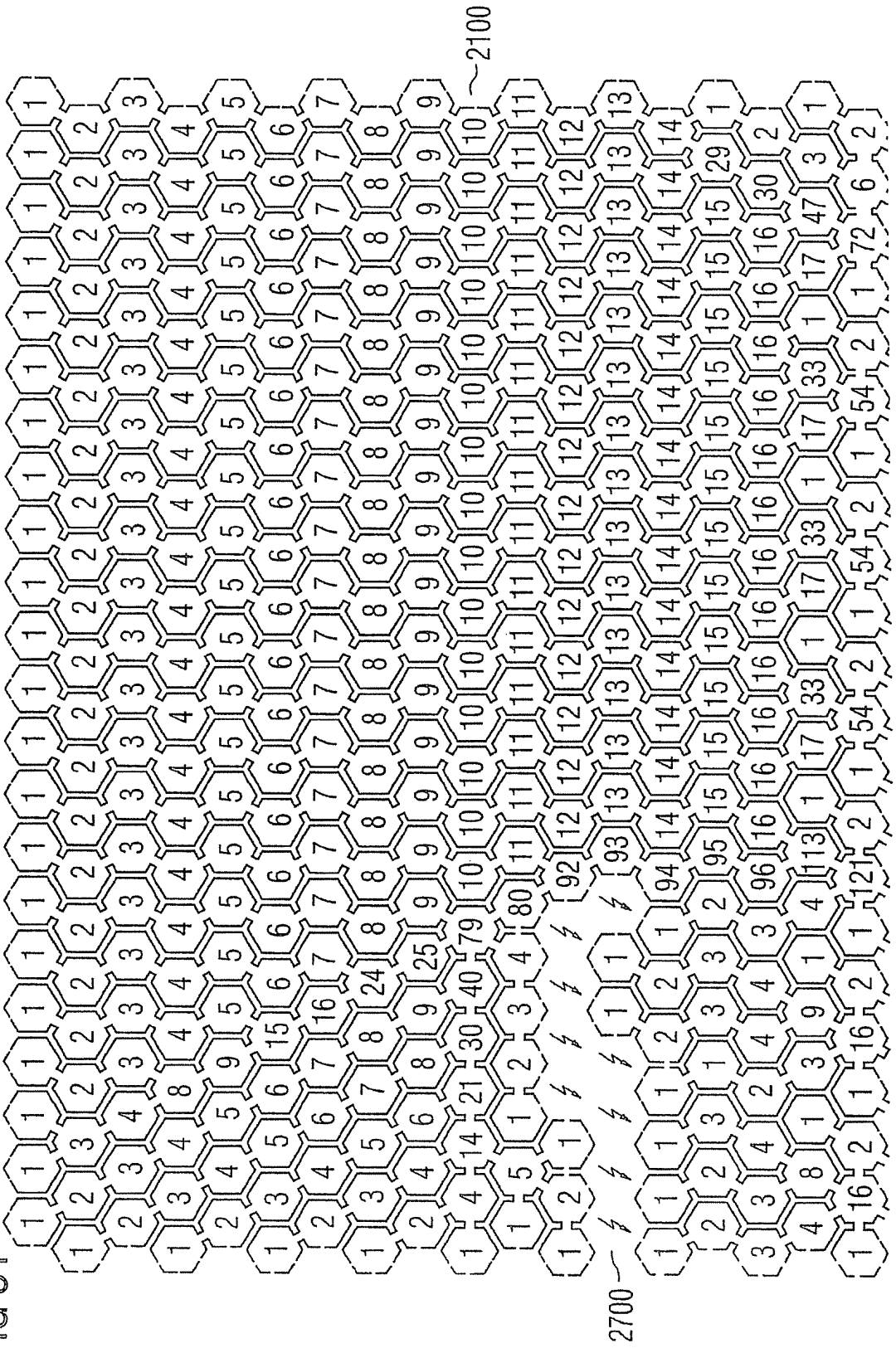


FIG 31

Nummer: DE 101 58 784 A1
Int. Cl.[7]: G 06 F 15/173
Offenlegungstag: 7. August 2003

FIG 32

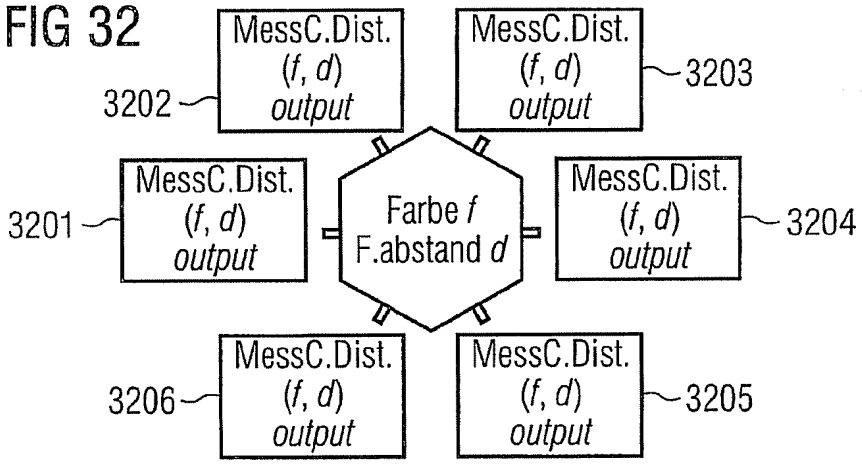


FIG 33

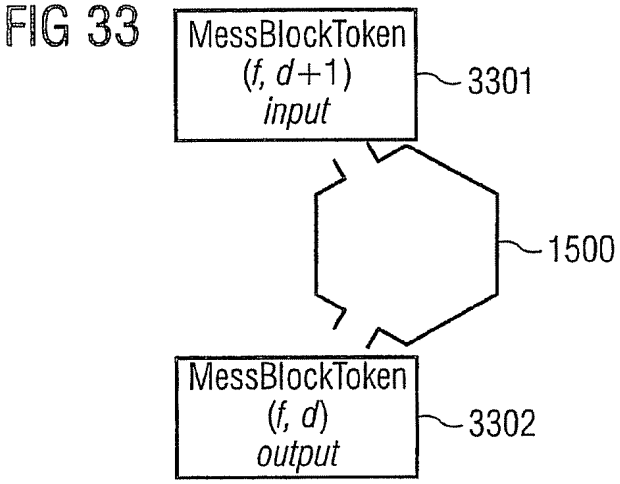
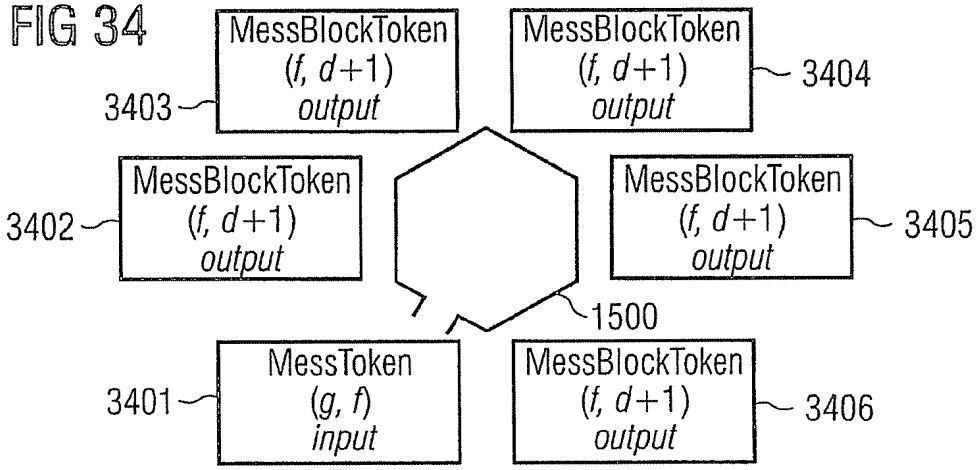


FIG 34



Nummer: DE 101 58 784 A1
Int. Cl.[7]: G 06 F 15/173
Offenlegungstag: 7. August 2003

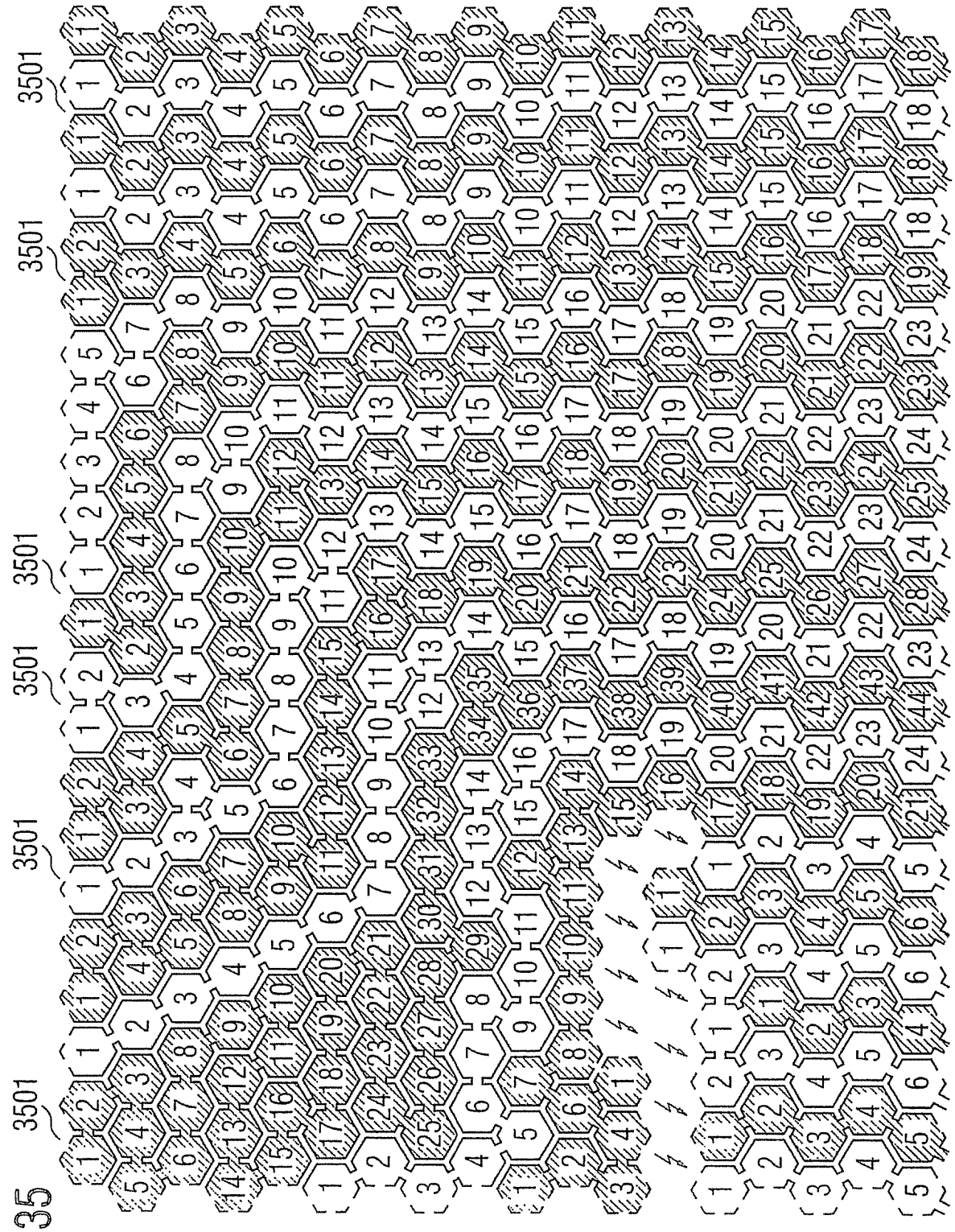


FIG 35

Nummer: DE 101 58 784 A1
Int. Cl.[7]: G 06 F 15/173
Offenlegungstag: 7. August 2003

FIG 36

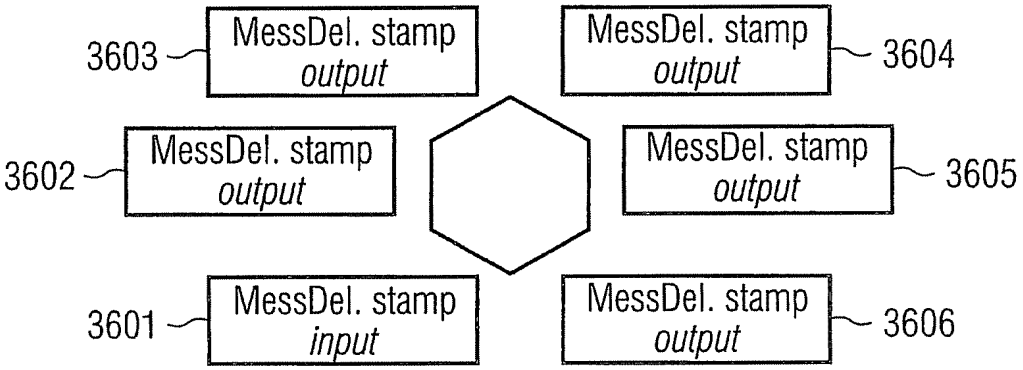


FIG 37

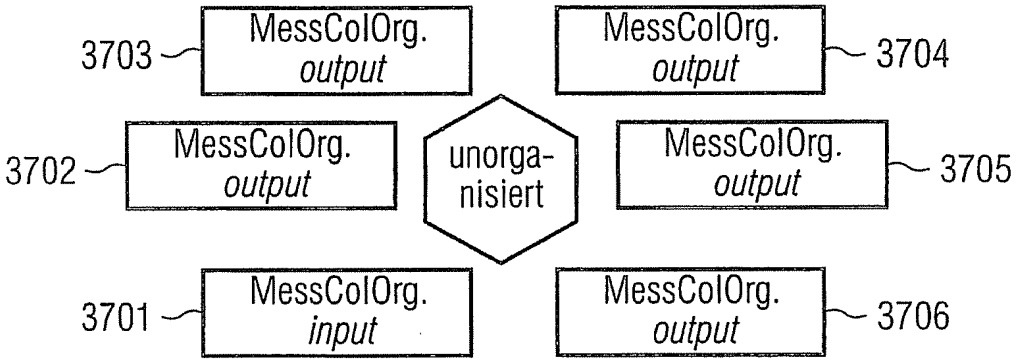
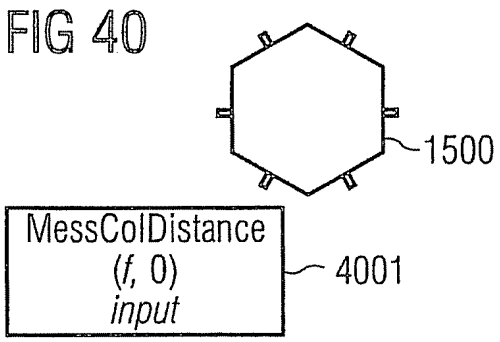


FIG 40



Nummer: DE 101 58 784 A1
Int. Cl.[7]: G 06 F 15/173
Offenlegungstag: 7. August 2003

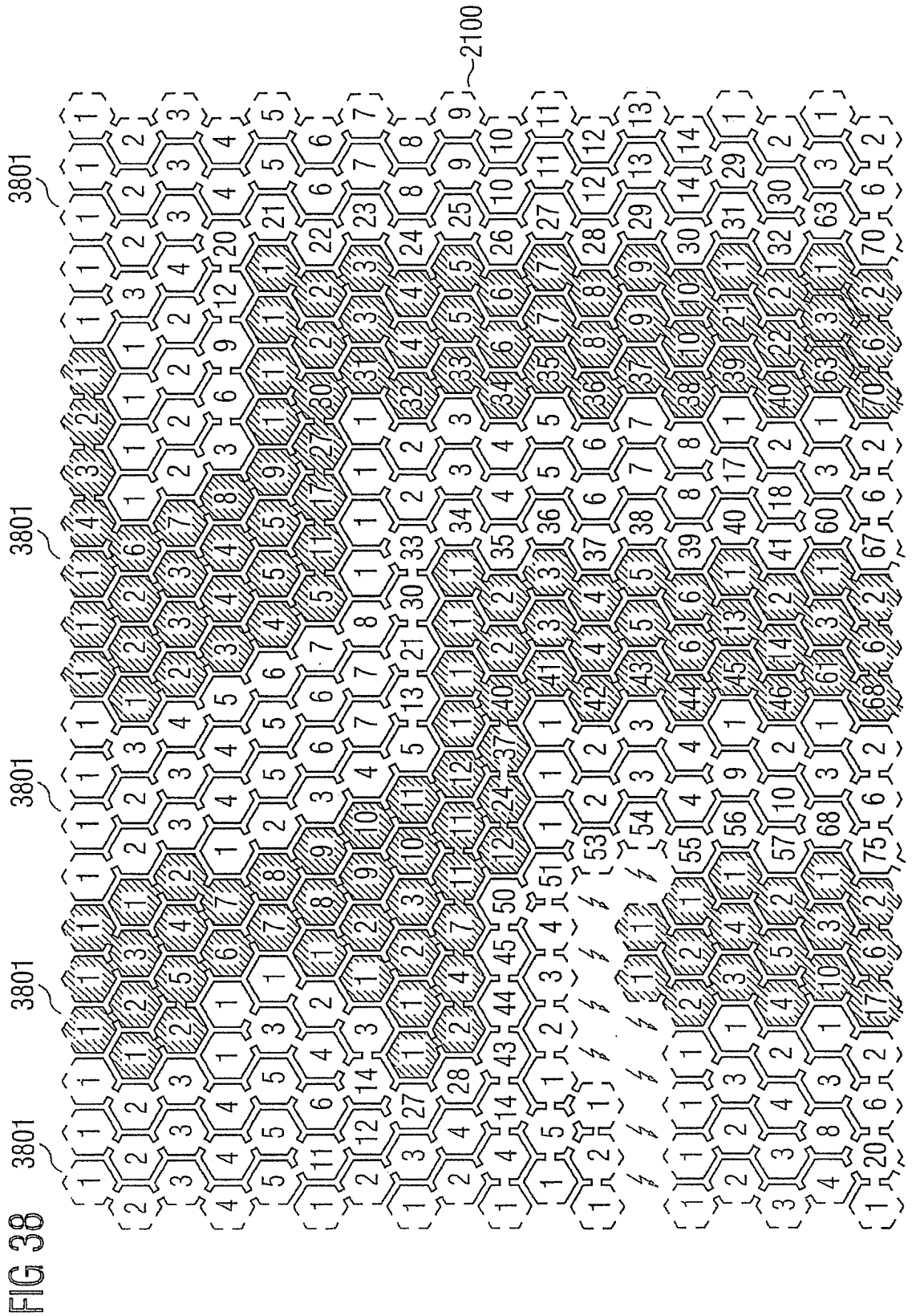


FIG 38

Nummer: DE 101 58 784 A1
Int. Cl.[7]: G 06 F 15/173
Offenlegungstag: 7. August 2003

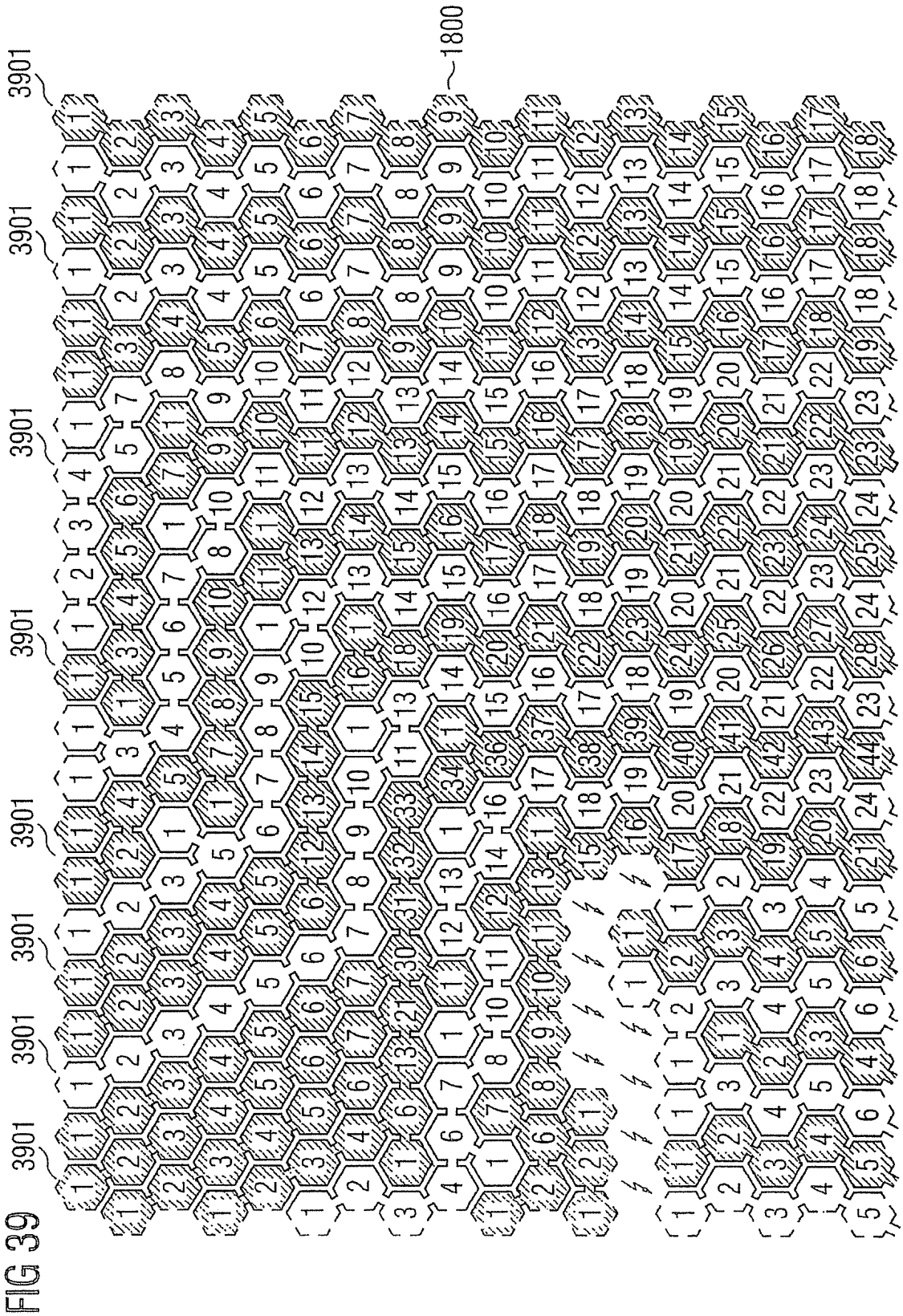


FIG 39

Nummer: DE 101 58 784 A1
Int. Cl.[7]: G 06 F 15/173
Offenlegungstag: 7. August 2003

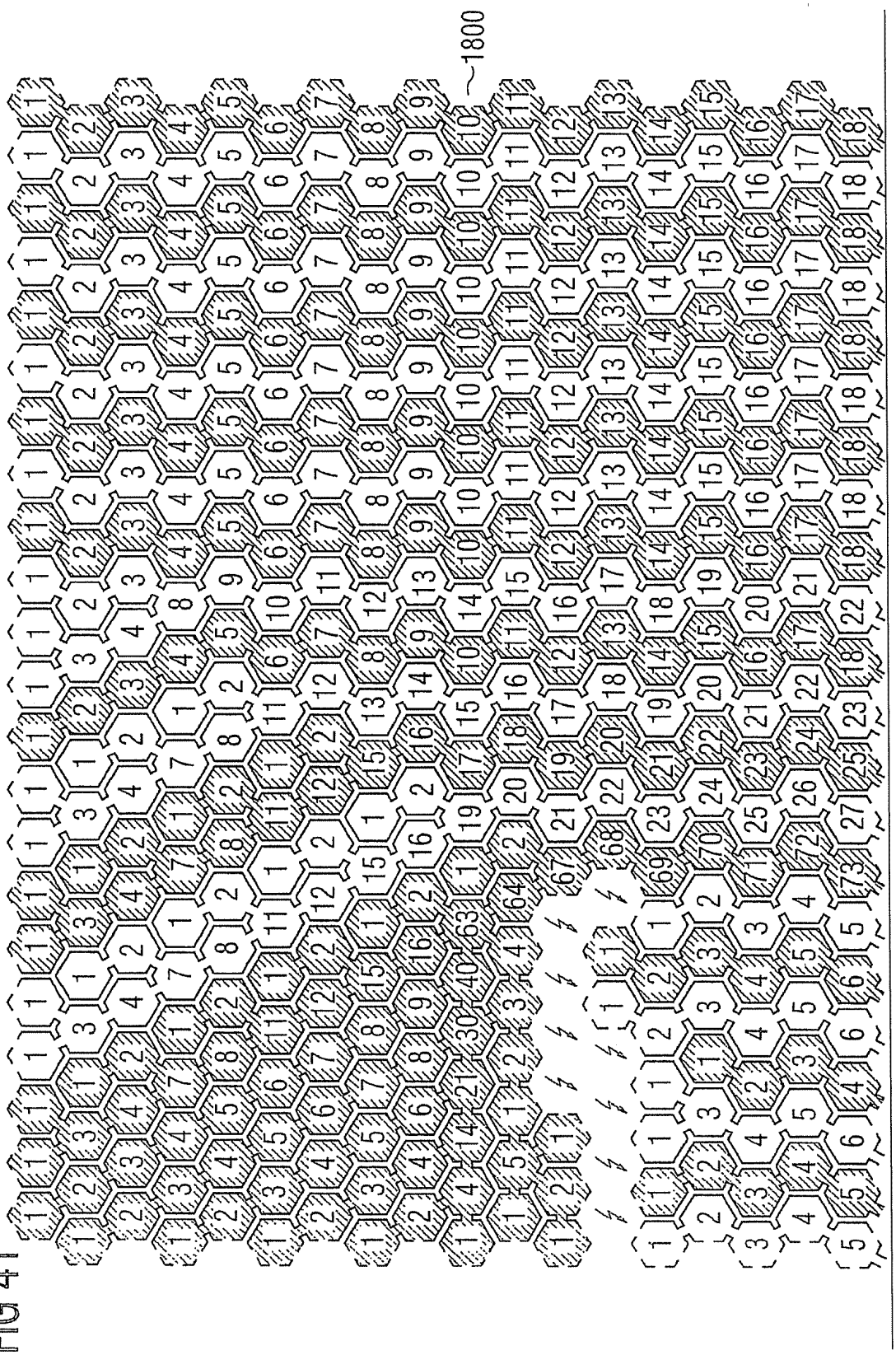


FIG 41

Nummer: **DE 101 58 784 A1**
Int. Cl.[7]: **G 06 F 15/173**
Offenlegungstag: 7. August 2003

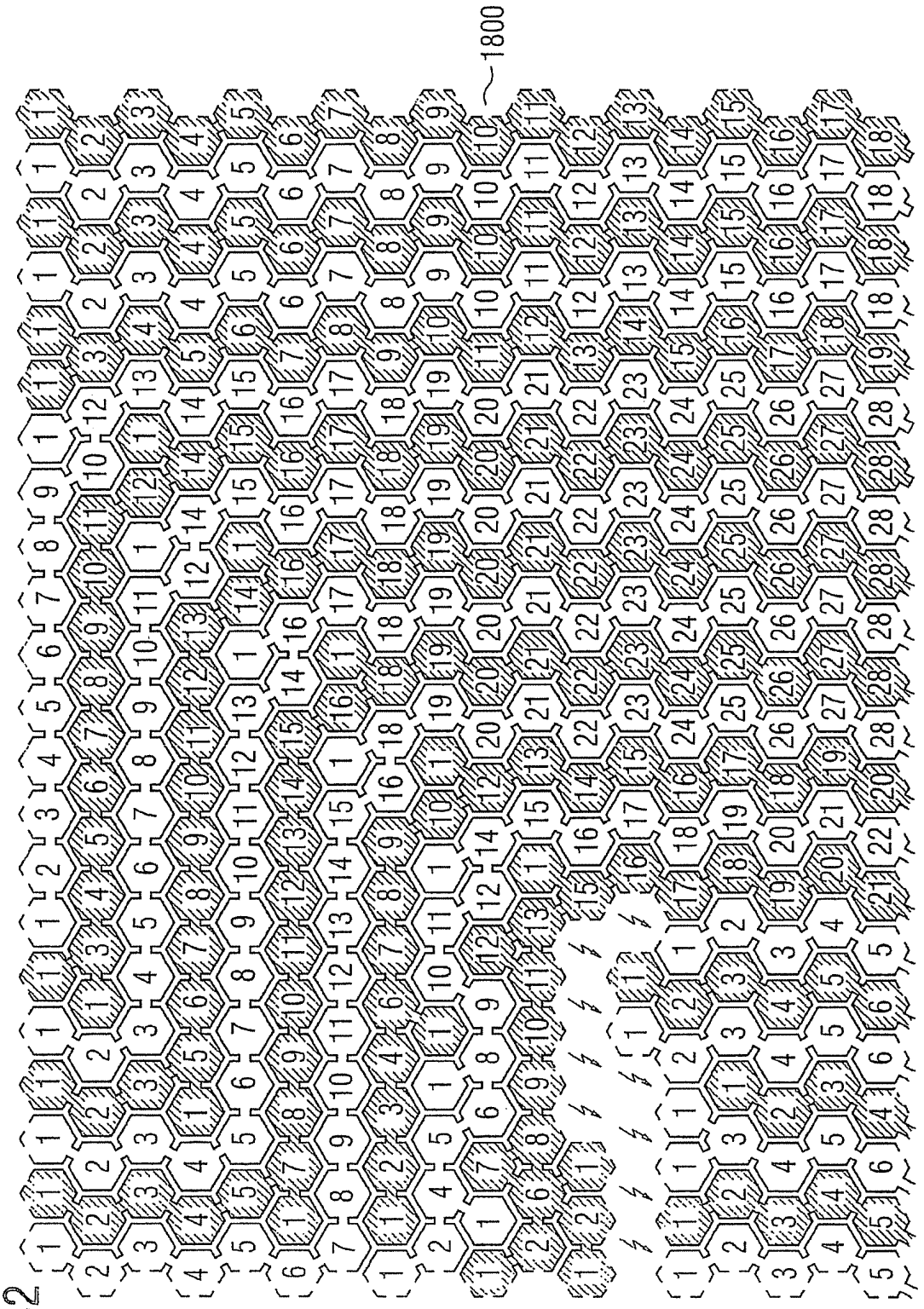


FIG 42

Nummer: DE 101 58 784 A1
Int. Cl.[7]: G 06 F 15/173
Offenlegungstag: 7. August 2003

FIG 43

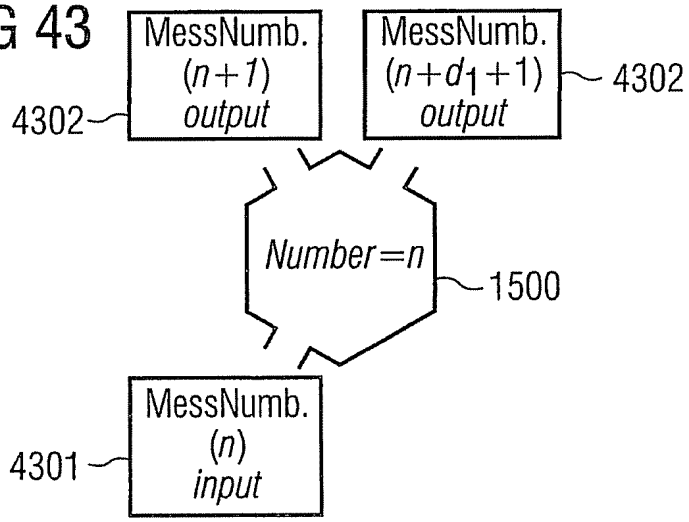
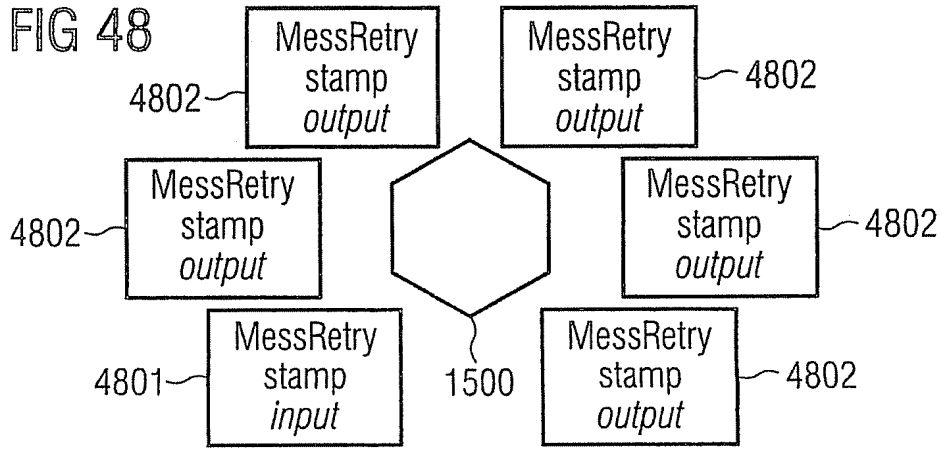


FIG 46

Beispiel: Routing-Tabelle Pixel Nr.123	
Nachrichtennummer *n*	Aktion
123	selbst Empfänger!
$124 \leq n \leq 135$	Versenden über Ausgang 1
$136 \leq n \leq 146$	Versenden über Ausgang 2
sonst	Fehler

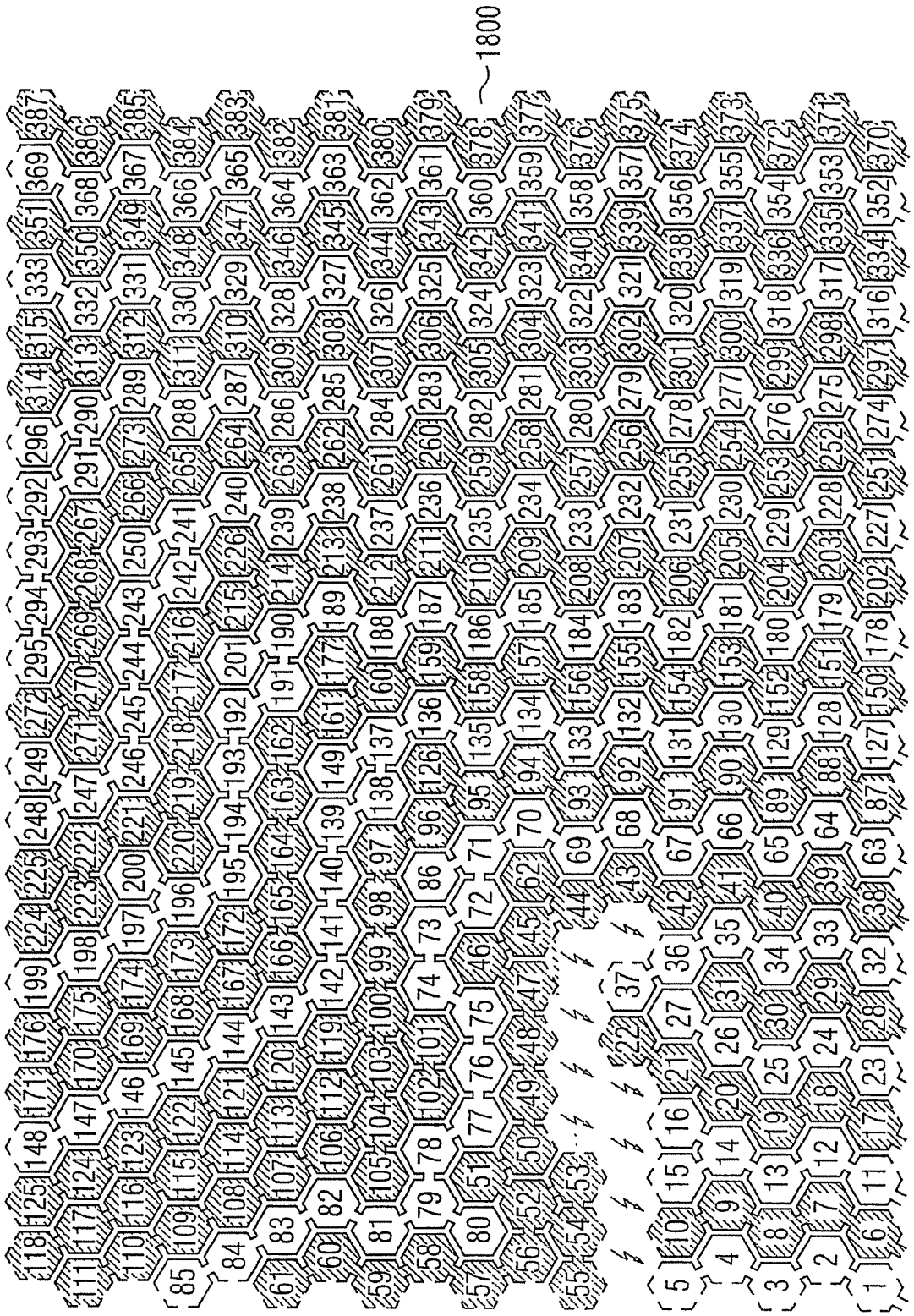
4600

FIG 48



Nummer: DE 101 58 784 A1
Int. Cl.[7]: G 06 F 15/173
Offenlegungstag: 7. August 2003

FIG 44



Nummer: DE 101 58 784 A1
Int. Cl.[7]: G 06 F 15/173
Offenlegungstag: 7. August 2003

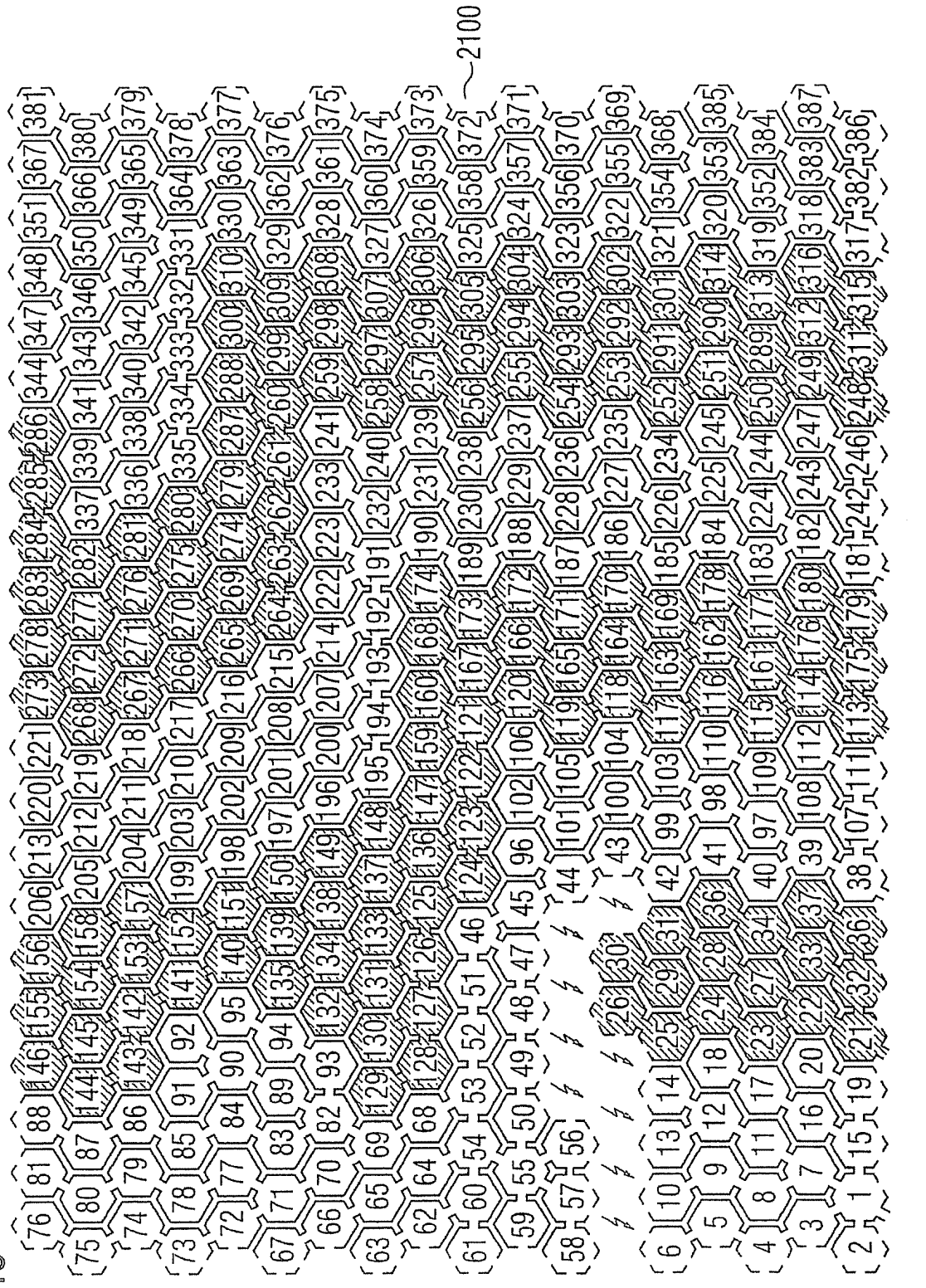


FIG 45

Nummer: **DE 101 58 784 A1**
Int. Cl.[7]: **G 06 F 15/173**
Offenlegungstag: 7. August 2003

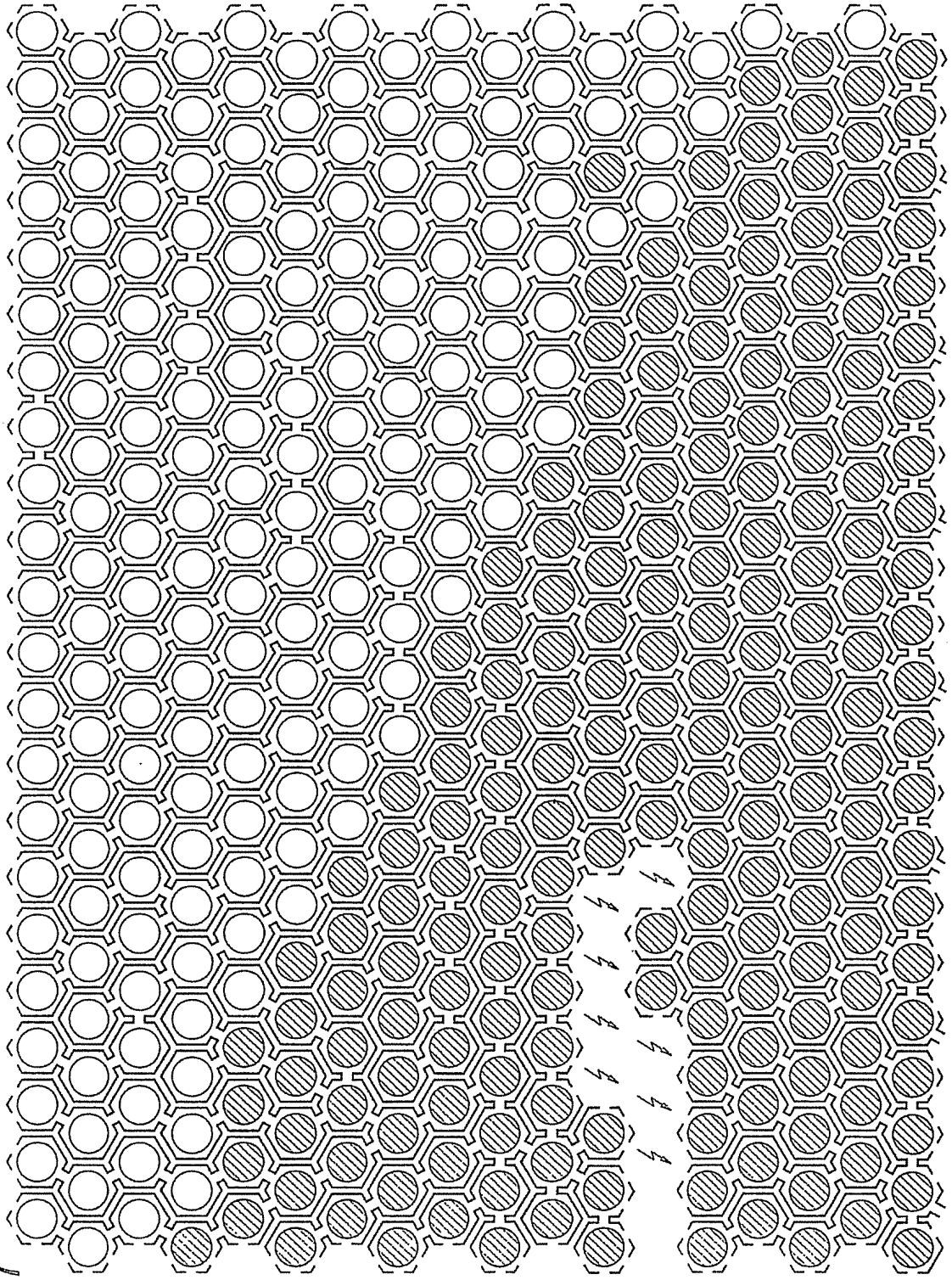


FIG 47

Nummer: DE 101 58 784 A1
Int. Cl.[7]: G 06 F 15/173
Offenlegungstag: 7. August 2003

FIG 49

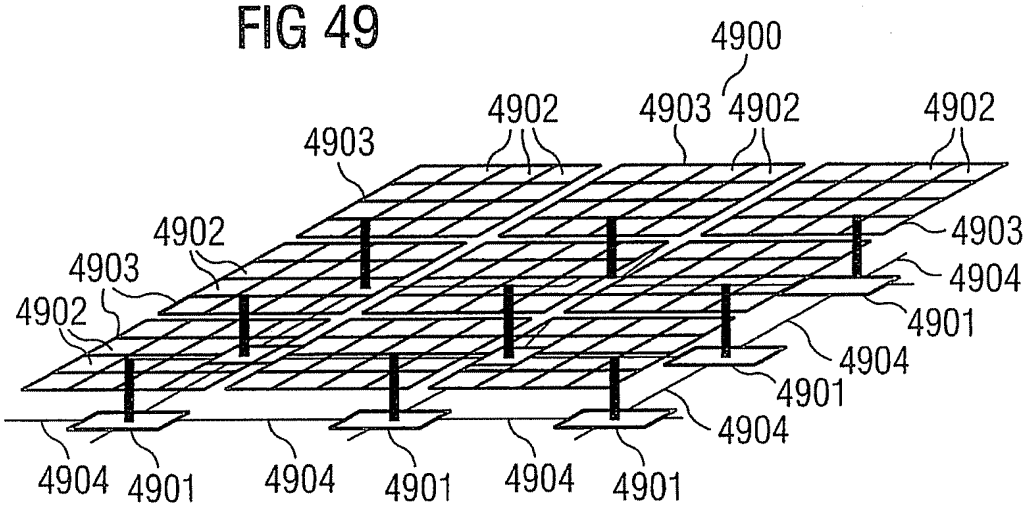
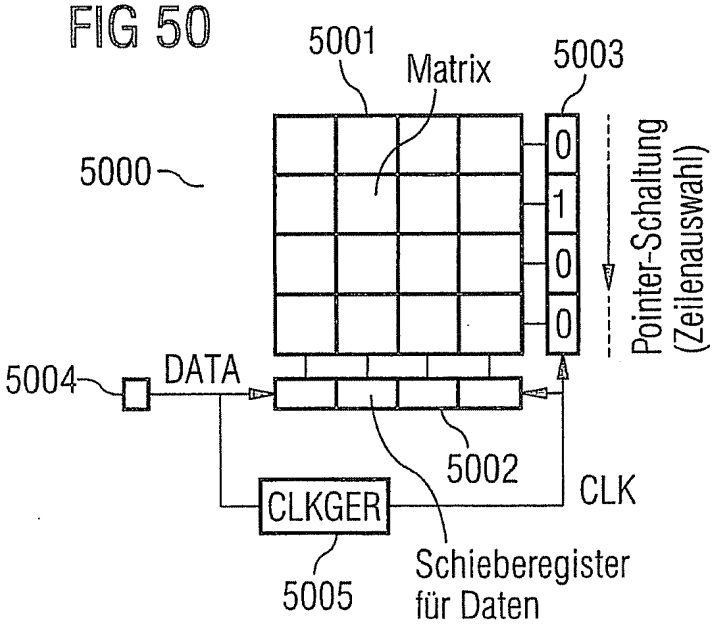


FIG 50



Nummer: DE 101 58 784 A1
Int. Cl.[7]: G 06 F 15/173
Offenlegungstag: 7. August 2003

FIG 51

Nachricht	Parameter
MessBlock Token	Farbe, Farbabstand
MessChannel	-
MessColDistance	Farbe, Farbabstand
MessCollectinfo	Zeile, Spalte, Pixelnummer, Abstand, Farbabstand, Durchsatz
MessColOrganize	-
MessCountNodes	-
MessDeleteChannels	Stempel
MessDistance	Abstand
MessError	Pixelnummer, Verbindungsnummer
MessKoherenz	Richtung
MessNodesSize	Durchsatz
MessNumbering	Pixelnummer
MessOrganize	-
MessPosition	Zeile, Spalte
MessReset	-
MessRetry	Stempel
MessRGB	Pixelnummer, Rot, Grün, Blau
MessToken	Gewicht, Farbe